14,- DM Ös 120,- sfr 14,-

SONDERHEFT
CP/M - Floppys - Hardware

CPC Schneider INTERNATIONAL

Sonderheft
2/86



Floppy-Tips:

u.a.
- RELATIVE DATEI
- DISKMANAGER
- CPC DOS 3.0

Hardware:

Nachwuchs für die DDI-1
- 3,5''-Floppy am CPC

CP/M:

DATENTRANSFER
MIT IBM UND APPLE

TURN-KEY SYSTEM

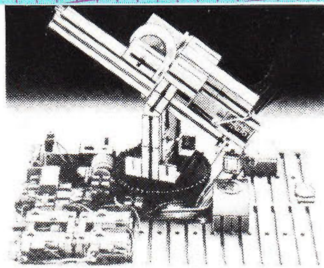
Software:

TIPS ZU WORDSTAR UND dBASE II

UNI-DATEI u.v.m.

Joyce: Mallard-Basic
Jetsam

Fischertechnik-computing

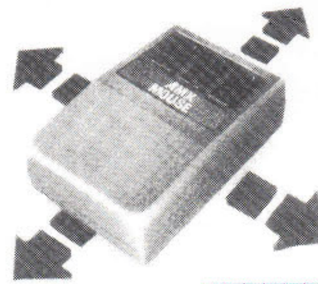


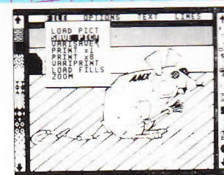
Neue Anwendungsmöglichkeiten für Ihren CPC 464 bieten die Fischertechnik-computing-Bausätze.

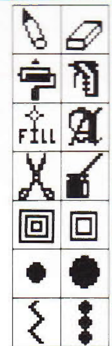
Simulation von Bewegungsabläufen und spielerisches Erlernen von Steuerungen mittels Computer.

	Bestell-Nr.	Preis
computing-Baukasten für 10 Modelle	821	199,– DM
Plotter/Scanner	822	449,– DM
Trainings-Roboter	823	499,– DM
Interface Schneider passend zu allen 3 Artikeln incl. Software auf Cassette	824	249,– DM
computing-Netzgerät (zum Betrieb der 3 Artikel erforderlich)	825	59,95 DM

AMX-MAUS





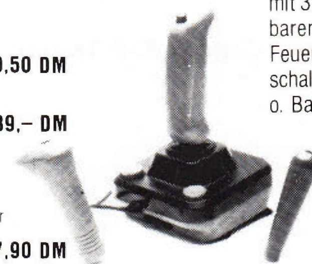


Die 1. Maus für den CPC

Best.-Nr. 809 **148,– DM**

NEU **Profi-Kunststoffhüllen**
leer, für Ihre Programmkassetten und Disketten im Buchformat

Best.-Nr. 837 für Cass. + 3" Disk. 12 Stck. **48,– DM**
Best.-Nr. 838 für Disk. 5¼" 12 Stck. **48,– DM**



Bildschirmvorsatz

bringt besseres Bild, saubere Entspiegelung!

für Grünmonitor
Best.-Nr. 6530 **99,– DM**

für Farbmonitor
Best.-Nr. 6540 **129,– DM**



Datenrecorder MC 3810

für CPC 664/6128

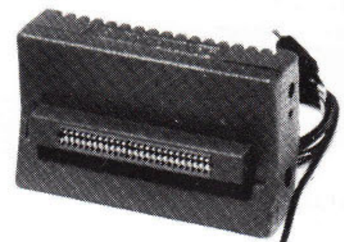
anschlußfertig inkl. Kabel und Netzteil.

Best-Nr. 836 **78,– DM**

Ohne Abbildung.
Schneider-Monitor-Verlängerungskabel
für CPC 464
Best.-Nr. 826 **19,50 DM**
für CPC 664 + 6128
Best.-Nr. 827 **39,– DM**

Joystick-Adapter
Zum Anschluß von 2 Standard-Joysticks an Schneider Computer
Best.-Nr. 828 **17,90 DM**

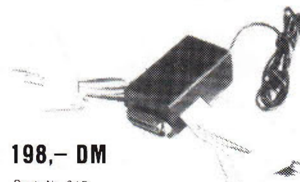
NEU **Joystick** Besonders stabil, mit 3 auswechselbaren Griffen, 2 Feuerknöpfen umschaltbar auf Stick o. Basis.
39,– DM



Staubschutzhaube

f. Grün-Monitor	816	37,50
f. Farb-Monitor	817	37,50
f. Drucker	818	22,50
f. CPC 464	819	22,50
f. Floppy	820	19,90
f. 664	–	22,50
f. 6128	–	22,50

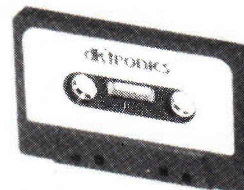
NEU **6510 Druckerinterface**
von Data Media für Schneider CPC 464/664
2 verschiedene Drucker können wahlweise betrieben werden. Steuerung über die Software, durch Eingabe des jeweiligen Druckercodes.
Mit eigener Stromversorgung.

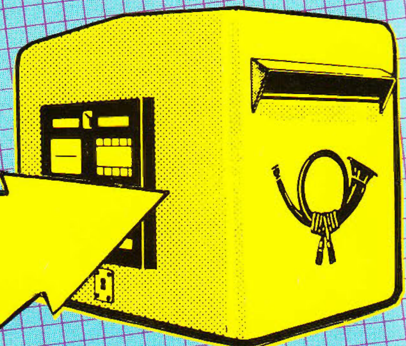


198,– DM
Best.-Nr. 815
für 3 Drucker **248,– DM**
Best.-Nr. 6511

dk'tronics Lightpen für CPC 464 und 664

Das Werkzeug für den Grafikprogrammierer: Pixelweise Auflösung, volle Ausnutzung der Farb- und Grafikmöglichkeiten des CPC, auch für Grünmonitor, umfangreiches Hilfsprogramm mit Menuesteuerung über Pictogramme, dadurch kinderleichte Handhabung, kompatibel zu DDI-1 und MP1. Komplett mit Handbuch und Hardcopyroutine.





Best.-Nr. 839
nur 99,– DM

Ohne Abbildung.
Schneider-Stereo-Kabel
3,5 mm Klinken-Stecker/
5 pol. DIN-Stecker, 150 cm lang
Best.-Nr. 812 **16,50 DM**

3,5 mm Klinken-Stecker/
2 Chinch-Stecker, 150 cm lang
Best.-Nr. 813 **16,50 DM**



Druckerkabel
für Schneider-Drucker, 150 cm lang
Best.-Nr. 814 **79,– DM**

NEU **Diskettenbox für 3"-Disketten**

aus antistatischem, schlagfestem Kunststoff, mit transparentem, aufklappbarem und abnehmbaren Deckel mit 3 Stützplatten inkl. Sichtreitern, für 40 Disketten, nicht abschließbar.

Best.-Nr. 840 **44,– DM**
dito mit Schloß Best.-Nr. 841 **54,– DM**

Data Media MAILORDER

Bestellen Sie noch heute, Lieferung erfolgt umgehend! Zahlung per Vorkasse oder Nachnahme zuzügl. Porto- bzw. Nachnahmegebühr (Nachnahme ins Ausland ist nicht möglich).

Weitere Artikel in unserem Gesamtkatalog. Bitte anfordern (2,– DM für Rückporto beilegen).
Data Media GmbH -Mailorder- Ruhrallee 55, 4600 Dortmund, Tel.: (02 31) 12 50 71-3

Impressum

Schneider CPC International Sonderheft 2/86
erscheint in der
DMV Daten & Medien Verlagsges. mbH

Chefredakteur
Christian Widuch (verantwortlich)

Redaktion
Stefan Ritter (SR), Thomas Morgen (TM), Michael Ceol (MC), Michael Ebbrecht (ME)

Satz
Silvia Führer, Renate Wells

Gestaltung
Renate Wells, Gerd Köberich

Grafik/Illustration
Heinrich Stiller

Fotografie
Gerd Köberich

Anzeigenleitung
Wolfgang Schnell
Es gilt die Anzeigenpreisliste Nr. 2 vom 1.1.1986

Freie Mitarbeiter
G. Brinkmann, J. Gerkens, K.-P. Trenkel, M. Wegmann, G. Friedle, G. Trier, A. Bamberg, H. Kleiner, Dipl.-Ing. H. Scheruhn, W. Wantia, J. Weiß, G. Treusch, Dipl.-Volksw. H. Pick, H.J. Behrend, M. Lipka, M. Anton, T. Fippel, H. Jakstatt, A. Grass, V. Sörgel, H. Franke (HF)

Anschrift Verlag/Redaktion
DMV Daten & Medien Verlagsges. mbH
Postfach 250
3440 Eschwege
Telefon: 05651/8702

Vertrieb
Verlagsunion
Friedrich-Bergius-Straße 20
6200 Wiesbaden

Druck
Druckerei Jungfer, 3420 Herzberg

Bezugspreise
Schneider CPC International Sonderheft 2/86 erscheint einmal jährlich.
Einzelpreis DM 14,-/sfr .14,-/ÖS 120,-



Liebe Leser

Nachdem unser erstes Sonderheft bei den Schneider-Anwendern einen hervorragenden Anklang gefunden hat, wurden Rufe nach einer Lektüre für die vielen Laufwerksbenutzer laut.

So ist auch unser Untertitel »CP/M – Floppys – Hardware« exakt auf diesen Themenkreis zugeschnitten. Das Ziel dieses Sonderheftes ist es, sowohl ein Nachschlagewerk als auch eine unbedingt notwendige Arbeitshilfe an die Hand zu geben. Hierzu gehört auch die komplette Literaturübersicht für die CPC-Rechner.

Erster Schwerpunkt dieses Heftes ist das schier unerschöpfliche Thema CP/M, zu dem wir Referenzkarten, Tips und Anwendungen geben. Da CP/M-Literatur meist nur dürftig oder kaum zu erhalten ist, dürfte dies einen besonderen Leckerbissen darstellen.

Im Hardwareteil sind besonders die Bastler angesprochen. Wie Sie Zweitlaufwerke für den CPC im Selbstbau erstellen können, erfahren Sie in dieser Sparte.

Neue, zusätzliche DOS-Befehle und jede Menge Tips, bringen eine effektive Nutzung der Floppy. Hierzu haben wir eine Reihe von Hilfsroutinen aufgeführt.

Natürlich enthält dieses Magazin auch eine semiprofessionelle Programmsammlung. Die zweite RAM-Bank wird durch die »Relative Dateiverwaltung« für den CPC 6128 ausgenutzt. Weitere interessante Programme sind z.B. der Terminplaner und der umfangreiche Disk-Monitor.

Daß die CPC's auch an Rechner wie Apple oder IBM Daten austauschen können, erfahren Sie auf den Seiten 68 bzw. 74.

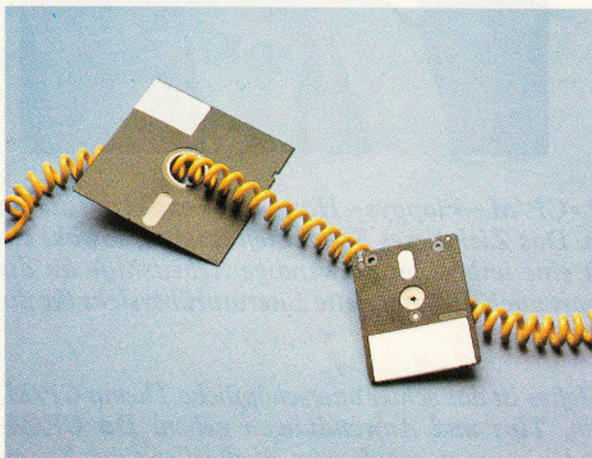
Alles in allem wieder eine Fülle von Informationen und Programmierhilfen, die Sie mit diesem Sonderheft erhalten! Dazu gibt es natürlich unseren bekannten Databox-Service, diesmal der Eigenheit der Themen entsprechend, natürlich nur als 3"-Diskettenversion.

Herzlichst Ihr

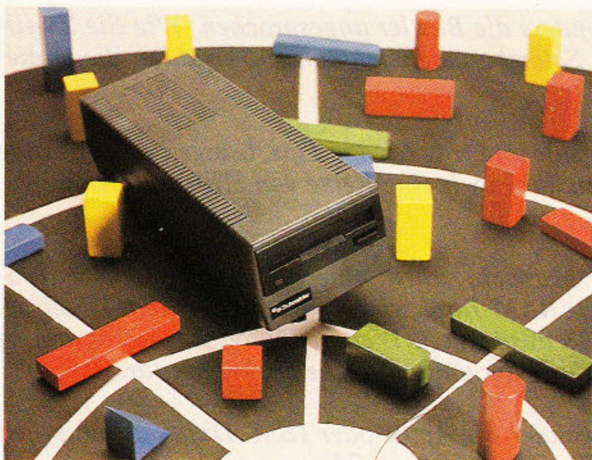
Christian Widuch
Chefredakteur



CP/M ist nicht nur eines der verbreitetsten Betriebssysteme überhaupt, sondern auch eines der universellsten. Viele Tips zum Umgang mit CP/M (z.B. Autostart) sowie ausführliche Referenzkarten über Befehle und Dienstprogramme erwarten Sie. Aufgrund der noch immer mangelnden Literatur haben Sie so ein komplettes Nachschlagewerk für das ganze Jahr.



Da werden Träume wahr! Endlich ist es möglich, Schneider Computer mit »professionellen« PC's wie z.B. IBM oder Apple zu koppeln und Datenaustausch zu ermöglichen. Wie das im einzelnen funktioniert, erfahren Sie beim Durchlesen der großen Projektserie »Kopplungen«.



Tips und Anwendungen rund um die Diskettenlaufwerke sind ein großer Schwerpunkt unseres Sonderheftes. U.a. präsentieren wir das CPC DOS 3.0, das viele neue Befehle zur Verfügung stellt und Ihnen hilft, die Floppy noch optimaler nutzen zu können.



In unserer großen Hardware-Rubrik gibt es jede Menge Bauanleitungen, mit denen nicht nur die Einsatzbereiche der CPC-Computer wesentlich erweitert werden, sondern zusätzlich jede Menge Geld gespart werden kann.

Floppy

Tips + Tricks

Software

Hardware

CP/M

Joyce

Allgemeines

Marktübersicht:D

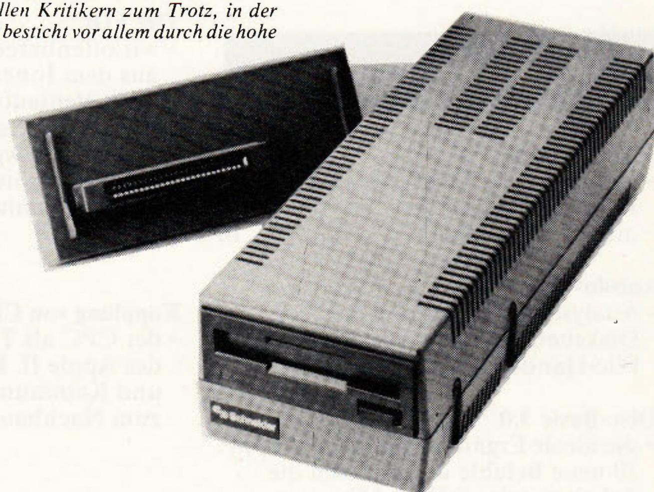


Im folgenden haben wir eine Marktübersicht der verschiedenen Diskettenstationen mit eigenem Controller zusammengestellt. Zu jeder Floppy finden Sie die entsprechenden technischen Daten sowie eine Kurzbeschreibung. Bei der Preisangabe ist zu beachten, daß wir von der kleinstmöglichen Gerätekonfiguration ausgegangen sind.

Zeichenerklärung:

●●● - gut
●● - befriedigend
● - verbesserungsbedürftig

Die Schneider DDI-1 im 3"-Format ist nach wie vor das Standardlaufwerk für die CPC's. Das seltene 3"-Format hat sich, allen Kritikern zum Trotz, in der Praxis bewährt und besticht vor allem durch die hohe Geschwindigkeit.

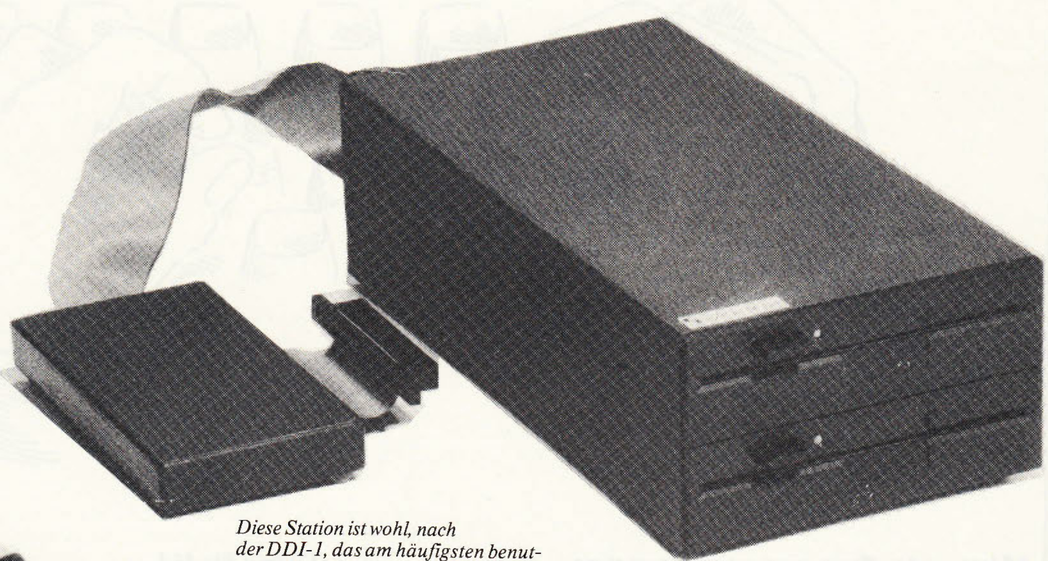


Typ	DDI-1	F1-S	111-S	FDD	DSB-35
Format	3"	5 ¼"	3,5"	5 ¼"	3,5"
Speicherkapazität	180 KB	704 KB	704 KB	780 KB	1 MB
Computer	464	464/664/6128	464/664/6128	464/664	464/664/6128
Tracks	40	80	80	40/80	80
Directory Einträge	64	64*	64*	72	99*
Preis	798,-	1198,-	998,-	1598,-	798,-
Handbuch	●●	●●●	●●●	●	●●
DOS	Amsdos	VDOS 2.0	VDOS 2.0	FDOS	Amsdos plus DSB51 Zusatzkontroller*
Maximal erweiterbar bis	2	3	3	4	2
Vertrieb	Schneider	Vortex	Vortex	Data Media	Data Service Cumana
Sonstiges	Logo, CP/M 2.2 werden mitgeliefert	CP/M wird mitgeliefert. *Mit Progr. PARA läßt sich die Anzahl erhöhen	CP/M wird mitgeliefert. *Mit Progr. PARA läßt sich die Anzahl erhöhen	CP/M wird mitgeliefert	DSB51 System

iskettenlaufwerke



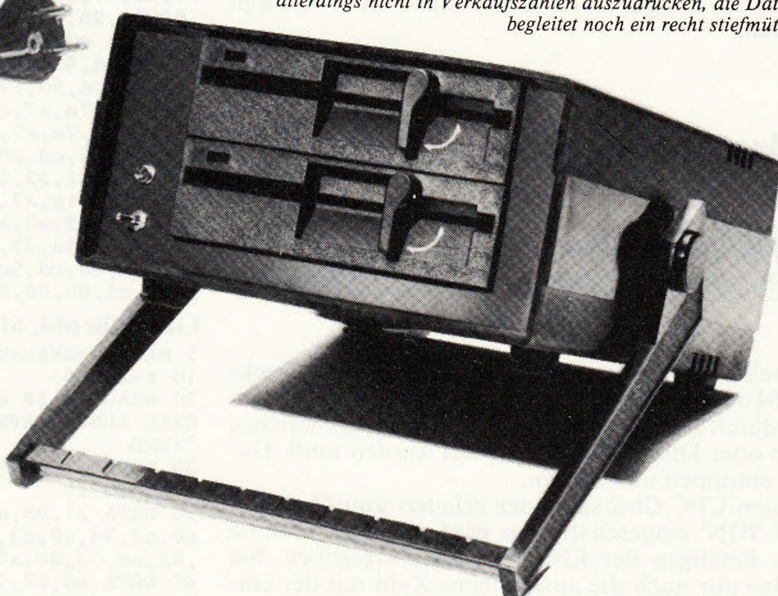
Die SSS.3.0 von Cumana ist ein preiswertes Zweitlaufwerk für alle CPC-Geräte.



Diese Station ist wohl, nach der DDI-1, das am häufigsten benutzte Speichermedium. Das 5 ¼"-Format hat eine große Speicherkapazität und die Datenträger sind wesentlich günstiger als die 3"-Disketten.



Ein weiteres Laufwerk im 5 ¼"-Format kommt von Cumana. Mit dem neuen Betriebssystem DSB-51 werden hervorragende Leistungen erzielt, eine weitere Alternative zur DDI-1.

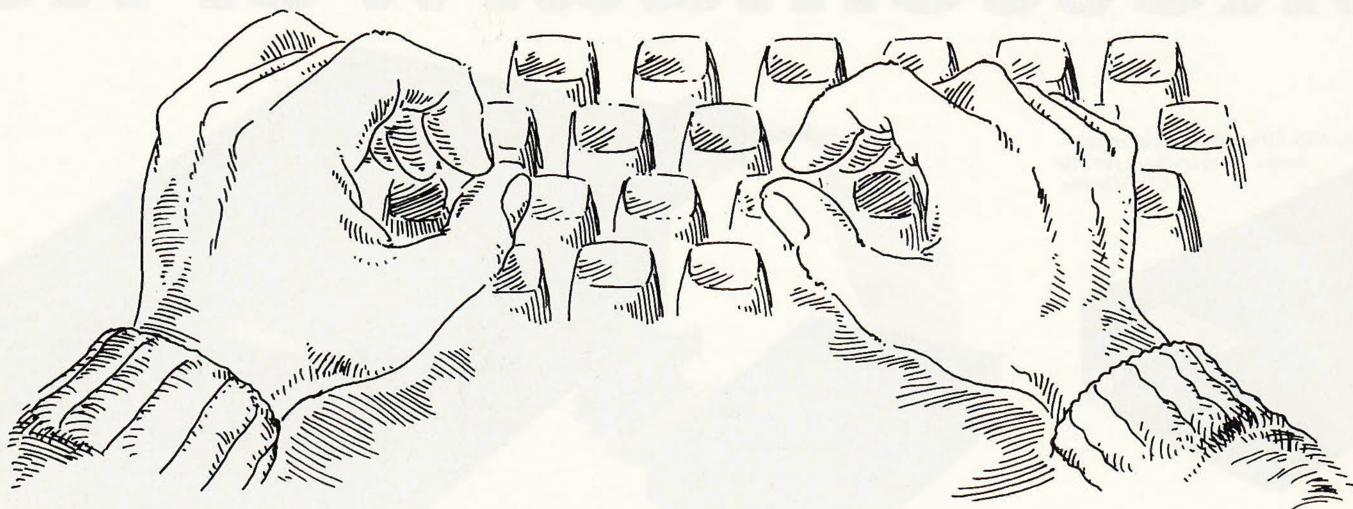


Die Data Media-Floppy sorgte aufgrund der hohen Speicherkapazität und der 40/80-Umschaltung für einiges Aufsehen. Die Erwartungen scheinen sich allerdings nicht in Verkaufszahlen auszudrücken, die Data Media-Station begleitet noch ein recht stiefmütterliches Dasein.

SSS.3.0	DSB 51-S
3"	5 ¼"
180 KB	1 MB
464/664/6128	464/664
40	40/80
64	99
459,-	848,-
-*	●●
Amsdos	Amsdos plus DSB51 Zusatzkontroller
2	2
Cumana	Data Service Cumana
Nur als Zweitlaufwerk	DSB51 System



Von Cumana ist ebenfalls ein 3,5"-Laufwerk erhältlich, das mit dem DSB-51-Betriebssystem von Data Service läuft. Große Leistung bei niedrigem Preis, das sind die Vorteile der DSB-35.



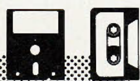
Hinweis-Programmservice

Zu den bekannten Symbolen der Rechnertypen sind noch Piktogramme für Kassetten- bzw. Diskettenbetrieb hinzugekommen.
Es ist nämlich ohne weiteres möglich, daß ein Programm auf dem 664 oder 6128 funktioniert, zum einwandfreien Arbeiten jedoch einen Kassettenrekorder benötigt.
In diesem Heft finden Sie auch wieder die Prüfsummen für den CPC-Checksummer. Denken Sie daran, daß diese Prüfsummen (in eckigen Klammern) nicht mit eingetippt werden dürfen.

(TM)

Checksummer für alle CPC

für 464-664-6128



Vorgehensweise: Zuerst entsprechende Version des Checksummers (464 oder 664/6128) eintippen und mit RUN starten. Es wird hierdurch ein Maschinenprogramm erzeugt, welches auf Kassette oder Diskette abgespeichert werden muß. Danach Lader eintippen und starten.
Haben Sie den CPC-Checksummer geladen und über den RSX-Befehl '|ON' eingeschaltet, so wird eine Quersumme nach jedem Betätigen der ENTER-Taste ausgegeben. Sie brauchen also nur noch die ausgegebene Zahl mit der entsprechenden im Heft zu vergleichen. Stimmen die beiden Werte nicht überein, haben Sie sich in der Zeile vertippt und können sie sofort korrigieren. Der Checksummer existiert in zwei Versionen. Die Versionen für 664 und 6128 sind identisch. Bei einigen 664-Versionen wurde jedoch ein ROM der Version 1 verwendet. Bei diesem ROM kann es zu Schwierigkeiten kommen, deren Grund uns noch nicht bekannt ist. Sobald wir den Fehler beheben können, werden wir Sie davon in Kenntnis setzen. Außerdem stellt der Checksummer noch den Befehl '|OFF' bereit, mit dem die Kontrollsummenausgabe abgeschaltet werden kann. 464-Benutzer haben auch noch die Möglichkeit, sich mit '|CHECK, Parameter' ein komplettes Listing mit Prüfsummen auf Bildschirm oder Drucker ausgeben zu lassen. 'Parameter' spezifiziert hierbei den Ausgabekanal. Es empfiehlt sich, mit WIDTH vorher die Druckbreite festzulegen.
Der |Check-Befehl gilt bisher nur für den CPC 464.

Listing für 464

```
5 REM checksummer version :464
10 t=&A000
20 READ a$:IF a$="xxx" THEN CLS:PRINT"press play":
CALL &BB06:SAVE"checkmc",b,&A000,&108:PRINT"fertig
":END
30 POKE t,VAL("&"+a$):t=t+1
40 GOTO 20
50 DATA 21,09,a0,01,0d,a0,c3,d1,bc,00,00,00,00,18,
a0,c3,2c,a0,c3,31,a0,c3,46,a0,4f,ce,4f,46,c6,43,48
,45,43,cb,00,00,00,00,cf,98,aa,c3,a3,a0,21
60 DATA 29,a0,18,03,21,26,a0,a7,28,06,cd,00,b9,c3,
96,dd,11,3a,bd,01,03,00,ed,b0,c9,4f,cd,00,b9,0d,28
,08,0d,20,ea,dd,7e,02,18,04,7b,11,01,00,cd,a2,c1
70 DATA cd,a3,e7,4e,23,46,2b,78,b1,c8,cd,3c,c4,e5,
09,e3,cd,63,e1,cd,75,a0,e1,18,ea,e5,cd,b8,a0,e3,cd
,93,a0,cd,96,f2,e3,cd,f0,a0,cd,4e,c3,e1
80 DATA 7e,a7,c8,cd,93,a0,cd,4e,c3,18,f5,3a,24,ac,
d6,08,47,7e,a7,c8,cd,45,e1,23,10,f7,c9,cd,26,a0,f5
,c5,d5,e5,cd,b8,a0,22,24,a0,cd,f0,a0,e1,d1,c1,f1,c
9,eb,1b,af,32,23,a0,47,67,6f,13,1a,d6,30,38,04,fe,
0a,38,f6,1a,a7,c8,4f,3a
90 DATA 23,a0,a7,79,20,07,eb,cd,41,dd,eb,c8,4f,fe,
22,20,08,3a,23,a0,2f,32,23,a0,79,ad,07,6f,09,13,18
,db,3e,20,cd,5c,c3,3e,5b,cd,5c,c3,cd,79,ee,3e,5d,c
3,5c,c3,00,00,00,00,00,00,00,00,00,xxx
```

Listing für 664, 6128

```
5 REM checksummer version :664/6128
10 t=&A000
20 READ a$:IF a$="xxx" THEN CLS:PRINT"press play":
CALL &BB06:SAVE"checkmc",b,&A000,&108:PRINT"fertig
":END
30 POKE t,VAL("&"+a$):t=t+1
40 GOTO 20
50 DATA 21,09,a0,01,0d,a0,c3,d1,bc,00,00,00,00,15,
a0,c3,24,a0,c3,29,a0,4f,ce,4f,46,c6,00,00,00,00,cf
,02,ac,c3,9b,a0,21,21,a0,18,03,21,1e
60 DATA a0,a7,28,06,cd,00,b9,c3,00,dd,11,5e,bd,01,
03,00,ed,b0,c9,4f,cd,00,b9,0d,28,08,0d,20,ea,dd,7e
,02,18,04,7b,11,01,00,cd,a6,c1,cd,64,e8,4e,23,46,2
b,78,b1,c8,cd,72,c4,e5,09,e3,cd,54,e2,cd,6d,a0,e1,
18,ea
70 DATA e5,cd,b0,a0,e3,cd,8b,a0,cd,52,f3,e3,cd,e8,
a0,cd,98,c3,e1,7e,a7,c8,cd,8b,a0,cd,98,c3,18,f5,3a
,09,ac,d6,08,47,7e,a7,c8,cd,22,e2,23,10
80 DATA f7,c9,cd,1e,a0,f5,c5,d5,e5,cd,b0,a0,22,1c,
a0,cd,e8,a0,e1,d1,c1,f1,c9,eb,1b,af,32,1b,a0,47,67
,6f,13,1a,d6,30,38,04,fe,0a,38,f6,1a,a7,c8,4f,3a,1
b,a0
90 DATA a7,79,20,07,eb,cd,2e,de,eb,c8,4f,fe,22,20,
08,3a,1b,a0,2f,32,1b,a0,79,ad,07,6f,09,13,18,db,3e
,20,cd,a8,c3,3e,5b,cd,a8,c3,cd,44,ef,3e,5d,c3,a8,c
3,00,00,00,xxx
```

Lader für 464, 664, 6128

```
10 MEMORY &9FFF
20 LOAD"!checkmc",&A500
30 CALL &A500:|ON
```

Relative Datei

Das Standard-Basic der CPC Computer ermöglicht nur das Arbeiten mit sogenannten sequentiellen Dateien, die mit Hilfe von OPENIN bzw. OPENOUT eröffnet und dann der Reihe nach gelesen bzw. beschrieben werden können. Das Merkmal einer sequentiellen Datei ist also, daß erst auf den nächsten Datensatz zugegriffen werden kann, wenn alle vorangegangenen Datensätze bereits bearbeitet wurden.
Äquivalent hierzu sind die vom Basic her bekannten Befehle READ, DATA und RESTORE.
Die Organisation einer sequentiellen Datei erweist sich daher als äußerst unpraktisch, wenn es zu sortieren gilt oder die Möglichkeit des wahlfreien Zugriffs auf jeden Datensatz gegeben sein soll. Genau hier setzt die relative Dateiverwaltung an, denn sie erlaubt es, frei auf jeden Datensatz zuzugreifen. Jeder Datensatz hat eine Nummer, mit der man ihn ganz einfach anwählen kann. Das entspricht genau jenem Prinzip, welches bei Arrays Anwendung findet, daher lohnt sich der Einsatz der relativen Dateiverwaltung erst, wenn man große Datenmengen speichern will, die die Kapazität des freien RAM-Speichers überschreiten würden. Beispiele hierfür gibt es genug, z.B. die Schallplatten-, Musikkassetten-, Bierdeckel- oder Briefmarkensammlung. Da kommt man schnell auf Stückzahlen, bei denen man sich fragen muß, ob man eventuell den RAM-Speicher des CPC aufstocken oder ein paar der geliebten Sammelobjekte abschaffen sollte, um den Rest nun endlich mit dem Computer verwalten zu können. Doch hier bietet sich glücklicherweise eine vernünftige Alternative an, nämlich die relative Dateiverwaltung.
Hat man sich für diese Möglichkeit entschieden, ist es notwendig, den Aufbau der relativen Datei im voraus zu planen. Zuerst wird die Struktur eines Datensatzes festgelegt. Jeder Datensatz kann maximal acht Felder mit einer Länge von insgesamt nicht mehr als 512 Zeichen enthalten. Wenn wir als Beispiel die Schallplattensammlung heranziehen, so könnte ein Datensatz folgendes Aussehen haben:

Feld/Länge/Bezeichnung

1	20	Interpret bzw. Band
2	30	Titel
3	30	LP
4	5	Spieldauer

In Feld Nr. 1 wird der Name des Interpreten bzw. der Band eingetragen, das nächste Feld nimmt den Titel des Liedes auf, dann kommt der Name der LP und als letztes die Spieldauer. Für jedes Feld muß man die maximale Länge mit angeben. Die Bezeichnungen dienen später als Gedächtnisstützen, sie werden bei der Ein- und Ausgabe der einzelnen Datensätze jeweils über dem dazugehörigen Feld mit ausgedruckt.

Zum Schluß muß man dann noch die maximale Anzahl der zu reservierenden Datensätze angeben, wobei man genügend Platz für zukünftige Erweiterungen berücksichtigen sollte. Ist die von Ihnen gewünschte Anzahl jedoch größer als die vom Computer aufgrund der freien Diskettenkapazität errechnete Zahl, dann kann Ihnen auch die relative Dateiverwaltung nicht mehr weiterhelfen.

Zur Bedienung des Programmes:

Das Programm RELDAT ermöglicht den Umgang mit relativen Dateien mit Hilfe eines zehnzeiligen Menues. Sämtliche Menuepunkte erwarten, daß die Arbeitsdiskette sich im Laufwerk A befindet, d.h. unter anderem, der Benutzer wird nicht ständig darauf hingewiesen, die Diskette einzulegen und ferner ist das Arbeiten mit einem Zweitlaufwerk nicht möglich.

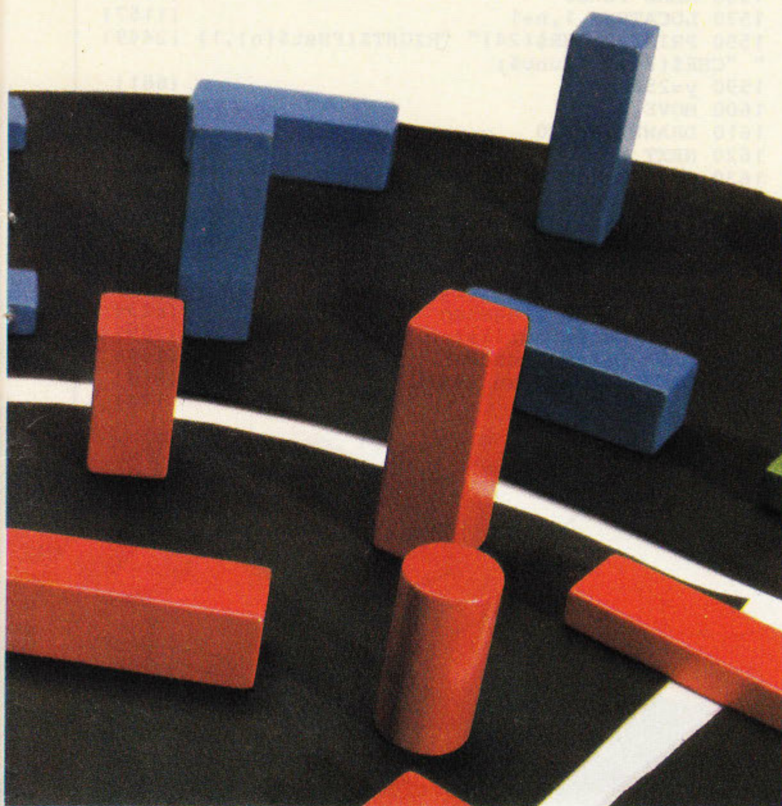
Die Menuepunkte im einzelnen:

1. Datei errichten
Definition der Dateistruktur, s.o.

2. Datei löschen
Von RELDAT erzeugte Dateien können hier mit der Eingabe des Namens wieder von der Diskette gelöscht werden.

3. Datei laden
Datei wird geöffnet, d.h. für die Bearbeitung vorbereitet.

4. Datei sichern
Datei wird geschlossen. Unbedingt notwendig vor Diskettenwechsel oder nach Abschluß der Arbeit mit einer geöffneten Datei.



5. Datei sortieren
Die Datei kann nach einem Feld in auf- bzw. absteigender Reihenfolge sortiert werden.

6. Directory zeigen
Der Übersichtlichkeit wegen, werden nur die von RELDAT erzeugten Dateien angezeigt.

7. Daten eingeben
Es können Datensätze an- bzw. eingefügt oder geändert werden. Die Eingabe des Kürzels "dt" bringt das aktuelle Datum in das gewünschte Feld

8. Daten auflisten
Man kann entweder alle Datensätze nacheinander ausgeben lassen, oder mit Hilfe einer Maske all diejenigen Datensätze finden, auf die diese Maske paßt.
Die Maske kann alles oder nur Teilstücke aus dem gewünschten Feld enthalten. Variable Teile mit konstanter Länge können mit Hilfe von Fragezeichen gleicher Anzahl, und variable Teile mit variabler Länge, an die sich ein Komma anschließt, können mit einem Sternchen ersetzt werden.

9. Daten löschen
Beliebige Datensätze können gelöscht werden. Die nachfolgenden Datensätze rücken in die frei gewordenen Plätze ein.

0. Zurück zum Basic
Eine geöffnete Datei wird selbständig geschlossen, danach wird das Programm abgebrochen.

Zu den Programmen:

RELDUMP erzeugt den Maschinencodeteil und speichert ihn auf einer Diskette ab, danach wird es nicht mehr benötigt.

RELDAT lädt den von RELDUMP erzeugten Maschinencode nach und initialisiert diesen, deshalb niemals das Programm ein zweites Mal mit RUN sondern nur mit GOTO 1030 starten.

Thomas Fippl

für 464-664-6128



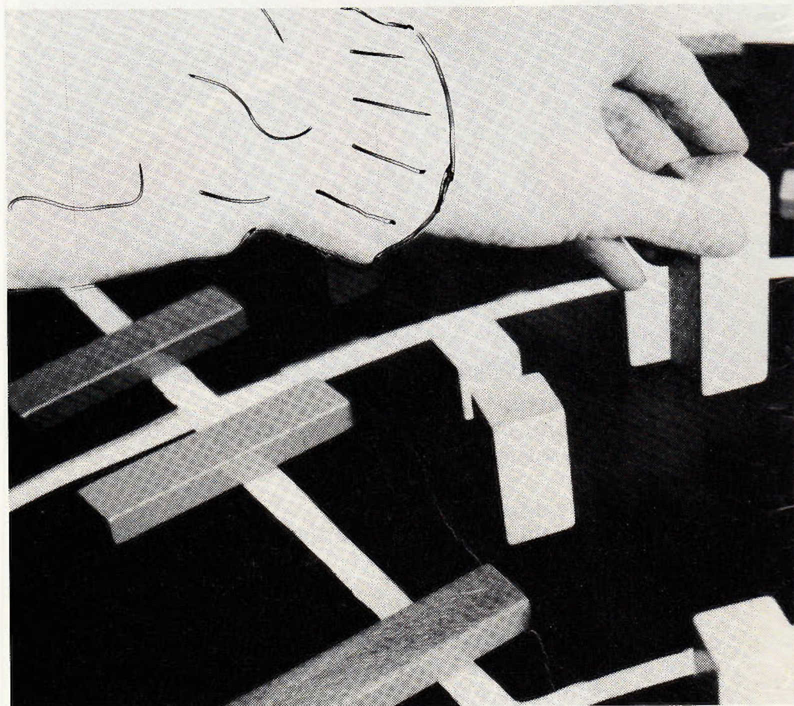
Listing Maschinencodelader

```
100 MEMORY &97FF:adr=HIMEM+1
110 FOR z=200 TO 292
120 sum=0:PRINT CHR$(30)z;
130 FOR n=0 TO 7:READ byte$
140 b=VAL("&"+byte$):POKE adr,b
150 sum=(sum XOR b)+b
160 adr=adr+1:NEXT:READ s
170 IF s<>sum THEN PRINT"Error":STOP
180 NEXT:SAVE"reldat",b,&9800,743
190 :
200 DATA 21,2d,98,df,4e,98,21,0f,450
201 DATA 98,01,13,98,c3,d1,bc,00,1420
202 DATA 00,00,00,51,98,c3,73,98,538
203 DATA c3,ea,98,c3,62,99,c3,89,1826
204 DATA 99,c3,fb,99,c3,4a,9a,c3,1588
205 DATA 7e,9a,c3,c4,9a,2a,00,b8,1462
206 DATA 0b,af,0f,0c,01,03,77,98,896
207 DATA 07,ee,98,07,66,99,07,8d,794
208 DATA 99,07,ff,99,07,4e,9a,07,1084
209 DATA 82,9a,07,c8,9a,07,0d,c6,1402
210 DATA 07,4f,50,45,ce,43,4c,4f,970
211 DATA 53,c5,46,49,4e,c4,52,45,856
212 DATA 43,4f,52,c4,47,45,d4,50,1260
213 DATA 55,d4,49,54,45,cd,45,4d,1212
214 DATA 49,d4,00,df,36,98,c9,cd,756
215 DATA c2,cd,3a,00,96,0e,01,a7,998
216 DATA c2,12,9a,eb,cd,ca,cd,11,1516
217 DATA e7,9e,cd,af,ce,21,e7,9a,1310
218 DATA 0e,00,11,19,a7,06,10,1a,366
219 DATA 13,a7,28,30,d5,c5,cd,23,1152
220 DATA 99,71,23,70,23,73,23,72,1000
221 DATA 23,c1,d1,0c,10,e9,3a,18,1016
222 DATA a7,fe,80,38,17,eb,86,77,1256
223 DATA dc,ad,d7,d5,fd,e5,d1,21,1022
224 DATA e7,9e,06,10,cd,92,d3,10,1050
225 DATA fb,e1,18,c6,79,f5,21,66,770
226 DATA c6,cd,13,99,f1,32,04,96,1244
227 DATA 21,1a,96,22,06,96,21,2c,428
228 DATA 96,22,08,96,3e,ff,32,00,910
229 DATA 96,c9,df,39,98,c9,cd,c3,1016
230 DATA cd,21,00,96,b6,ca,d7,99,1312
231 DATA af,77,23,77,23,b6,c4,55,784
232 DATA 99,21,00,00,22,0f,96,21,616
233 DATA 4e,c6,cd,13,99,cd,b6,d1,1358
234 DATA c3,10,9a,ed,4b,e7,9a,22,1160
235 DATA 21,99,21,00,96,50,1e,00,646
236 DATA c3,00,00,e5,6f,26,00,16,726
237 DATA 09,5c,29,42,a7,ed,52,2c,1012
238 DATA 30,02,19,2d,29,10,f5,cb,778
239 DATA 3c,cb,1d,3a,9f,a8,84,4f,1132
240 DATA 3a,9d,a8,85,47,e1,50,59,682
241 DATA 1c,7b,e6,0f,fe,0a,d8,3a,772
242 DATA 9f,a8,5f,14,c9,36,00,21,852
243 DATA 4e,c6,3a,11,96,cb,bf,c3,1304
244 DATA db,99,df,3c,98,c9,28,1b,1326
245 DATA cd,c2,cd,eb,cd,ca,cd,cd,1532
246 DATA 8d,da,cd,14,ce,cd,83,d6,1928
247 DATA cd,98,d6,0e,00,30,01,0d,574
248 DATA c3,12,9a,cd,c2,d8,4b,18,674
249 DATA f7,df,3f,98,c9,cd,c1,cd,1966
250 DATA 21,00,96,b6,28,41,23,77,788
251 DATA 23,d5,d5,a6,c4,55,99,21,1072
252 DATA e7,9e,d1,19,22,0c,96,26,906
253 DATA 00,dd,6e,02,29,11,e9,9a,1164
254 DATA 19,ed,5b,0f,96,22,0f,96,1138
255 DATA f6,ff,ed,52,28,02,e6,7f,1430
256 DATA e6,fe,2a,0a,96,d1,19,cb,1342
257 DATA 4c,28,01,3c,32,11,96,21,714
258 DATA 66,c6,cd,db,99,18,39,0e,716
259 DATA 02,18,37,22,21,99,f5,17,750
260 DATA 2a,0f,96,4e,23,56,e5,21,860
261 DATA e7,9e,d4,1e,99,e1,f1,1f,1386
262 DATA d0,23,4e,23,56,21,e7,a0,1288
263 DATA c3,1e,99,df,42,98,c9,cd,1058
264 DATA c1,cd,21,01,96,b6,28,cf,1438
265 DATA cd,1b,9a,71,23,73,23,72,912
266 DATA 0e,00,79,32,0e,96,c9,0e,956
267 DATA 03,18,f7,3a,05,96,dd,46,588
268 DATA 02,0e,05,e1,04,b8,38,ea,952
269 DATA e5,d5,21,1a,96,78,3d,87,1334
270 DATA 4f,06,00,09,4e,23,46,23,396
271 DATA 5e,23,56,eb,a7,ed,42,50,528
272 DATA 59,44,4d,2a,0c,96,19,eb,1136
273 DATA e1,c9,df,45,98,c9,cd,c1,1450
```

```
274 DATA cd,21,01,96,b6,ca,d7,99,1362
275 DATA 23,77,cd,1b,9a,79,96,23,964
276 DATA cd,f9,db,28,15,38,b0,f5,1406
277 DATA 47,79,90,c4,77,9a,c1,3e,1092
278 DATA 20,12,13,10,fc,18,99,06,664
279 DATA 00,4f,ed,b0,18,92,df,48,842
280 DATA 98,c9,cd,c1,cd,3a,00,96,1048
281 DATA a7,ca,d7,99,21,05,96,cb,1272
282 DATA 5e,0e,04,c2,12,9a,34,2a,376
283 DATA 06,96,4e,23,46,23,23,eb,1204
284 DATA 09,eb,72,2b,73,22,06,96,708
285 DATA 2a,08,96,dd,56,03,dd,5e,958
286 DATA 02,eb,7e,12,23,13,cd,f9,1222
287 DATA db,a7,c4,77,9a,ed,53,08,934
288 DATA 96,c3,10,9a,df,4b,98,c9,1476
289 DATA cd,c2,cd,47,3a,05,96,a7,1702
290 DATA ca,d7,99,2a,08,96,eb,1a,1458
291 DATA 77,13,4f,23,73,23,72,eb,834
292 DATA 09,22,08,96,c3,10,9a,00,848
```

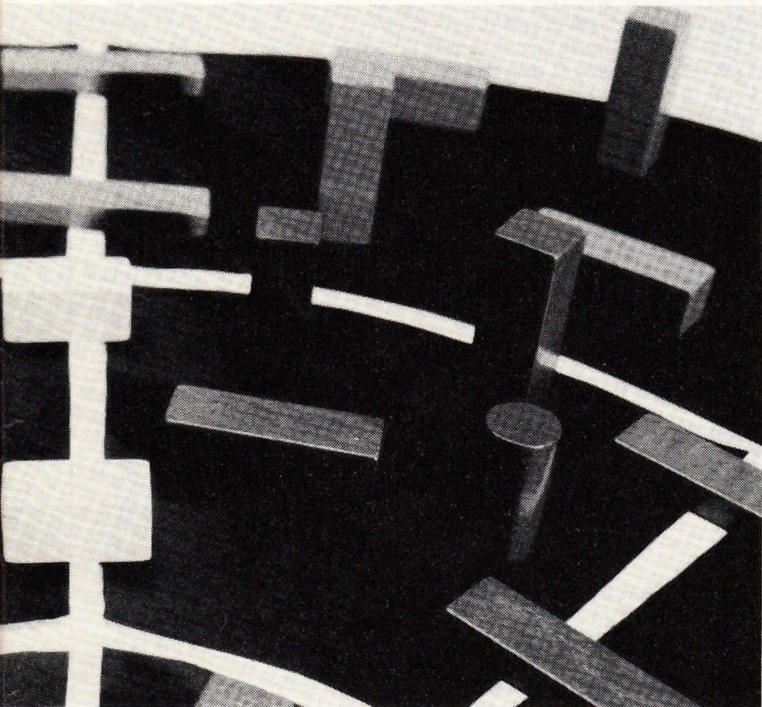
Programmlisting

```
100 '**************               [345]
110 '* Rel. Datei-  *             [361]
120 '*  verwaltung  *             [1958]
130 '**************               [345]
140 '*   (c)1986    *             [453]
150 '* by Th.Fippl  *             [570]
160 '*              *
170 '*  6336 Solms  *             [593]
180 '**************               [345]
190 '                             [117]
1000 |DISC:MEMORY &95FF           [2812]
1010 LOAD"reldat.bin",&9800       [3080]
1020 CALL &9800                   [477]
1030 ON BREAK GOSUB 1830          [561]
1040 SPEED KEY 15,1               [1264]
1050 DEFINT m,n,r:DEFSTR i        [658]
1060 DEF FNdt$(t,n)=RIGHT$("0"+FNst$(t),n) [1869]
1070 DEF FNst$(m)=MID$(STR$(m),2) [1058]
1080 DEF FNlo(m)=UNT(m)MOD 256    [403]
1090 DEF FNhi(m)=UNT(m)\256       [1106]
1100 DEF FNdp(m)=PEEK(m)+256*PEEK(m+1) [1612]
1110 bef1$=CHR$(&F2)+CHR$(&F3)+CHR$(&7F)+C [1554]
HR$(&10)
1120 bef2$=CHR$(&F0)+CHR$(&F1)+CHR$(&7F) [1777]
1130 header$=STRING$(128,CHR$(0)) [1553]
1140 blank$=SPACE$(255)           [2383]
1150 GOSUB 1280                   [859]
1160 '                            [117]
1170 '*** Functions-Auswahl ***   [2144]
1180 '                            [117]
1190 LOCATE#2,5,13                [899]
1200 PRINT#2,"Ihre Wahl: "CHR$(143)CHR$(8) [1647]
;
1210 i=INKEY$                     [560]
1220 IF i<"0"OR i>"9"GOTO 1210    [1421]
1230 PRINT#2,i;:CLS:PRINT         [2092]
1240 IF i="0"GOTO 6600            [622]
1250 ON VAL(i)GOSUB 3510,4130,4210,4450,45 [3737]
20,4940,5020,5850,6440
1260 PRINT CHR$(7);               [1175]
1270 GOTO 1160                    [307]
1280 '                            [117]
1290 '*** Bildschirm-Maske ***    [2722]
1300 '                            [117]
1310 MODE 2:BORDER 2              [1093]
1320 FOR n=1 TO 4                 [863]
1330 READ win$,x1,x2,y1,y2,pn     [1563]
1340 PAPER pn:PEN 1-pn            [1241]
1350 x=x2-x1+1                    [631]
1360 LOCATE x1-1,y1-1             [712]
1370 PRINT CHR$(150)CHR$(24)win$CHR$(24)ST [3876]
RING$(x-LEN(win$),CHR$(154))CHR$(156)
1380 FOR y=y1 TO y2               [1286]
1390 LOCATE x1-1,y                [636]
1400 PRINT CHR$(149)              [1036]
1410 LOCATE x2+1,y                [987]
1420 PRINT CHR$(149)              [1036]
1430 NEXT                         [350]
1440 LOCATE x1-1,y2+1             [704]
1450 PRINT CHR$(147)STRING$(x,CHR$(154))CH [2405]
R$(153);
1460 WINDOW x1,x2,y1,y2           [1672]
1470 WINDOW SWAP n                [1061]
1480 CLS#n:NEXT                   [627]
1490 WINDOW SWAP 1                [826]
1500 DATA Console,24,79,2,21,0    [1492]
1510 DATA Function,2,21,9,21,0    [1625]
1520 DATA Status,2,21,2,6,1       [1727]
1530 DATA Help,2,79,24,24,1       [1078]
1540 :                            [174]
1550 FOR n=1 TO 10                [1012]
1560 READ func$                   [697]
1570 LOCATE#2,1,n+1               [1157]
1580 PRINT#2,CHR$(24)" "RIGHT$(FNst$(n),1) [2449]
" "CHR$(24)" "func$;
1590 y=256-n*16                   [661]
1600 MOVE 8,y                     [635]
1610 DRAWR 24,0,0                 [550]
1620 NEXT                         [350]
1630 DATA Datei errichten         [1500]
1640 DATA Datei loeschen          [987]
1650 DATA Datei laden             [731]
1660 DATA Datei sichern           [1317]
1670 DATA Datei sortieren         [959]
1680 DATA Directory zeigen        [1219]
1690 DATA Daten eingeben          [823]
1700 DATA Daten auflisten         [2093]
1710 DATA Daten loeschen          [1898]
1720 DATA Zurueck zu BASIC        [910]
1730 :                            [174]
```

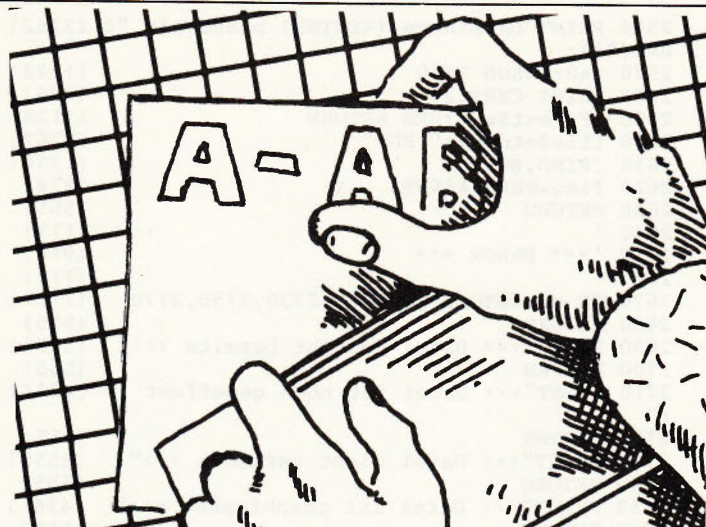


```
1740 INPUT#3,"Datum (Tag,Mon,Jahr)";t,m,j [2770]
1750 IF j<100 THEN j=j+1900       [1445]
1760 dat$=FNdt$(t,2)+"."+FNdt$(m,2)+"."+FN [3700]
dt$(j,4)
1770 CLS#3                        [371]
1780 PRINT#3,"Datum: "dat$        [1326]
1790 PRINT#3                      [316]
1800 PRINT#3,"Datei :"            [1080]
1810 PRINT#3,"belegt:"            [1041]
1820 PRINT#3,"frei  :";           [1302]
1830 RETURN                       [555]
1840 '                            [117]
1850 '*** INPUT String ***        [1237]
1860 '                            [117]
1870 back$=STRING$(t,".")         [800]
1880 text$=""                     [537]
1890 xp=POS(#0):xpo=xp            [1874]
1900 yp=VPOS(#0)                  [802]
1910 GOSUB 2180                   [855]
1920 :                            [174]
1930 GOSUB 2250                   [865]
1940 IF i=CHR$(13)GOTO 1970       [1252]
1950 GOSUB 2490                   [849]
1960 GOTO 1930                    [377]
1970 IF LOWER$(text$)="dt"THEN text$=dat$ [2469]
1980 PRINT:CLS#4                  [335]
1990 RETURN                       [555]
2000 '                            [117]
2010 '*** INPUT Zahl ***          [915]
2020 '                            [117]
2030 t=LEN(STR$(thi))-1           [1822]
2040 back$=STRING$(t,"_")         [892]
2050 xpo=POS(#0)                  [1519]
```

```
2060 yp=VPOS(#0)                              [802]
2070 GOSUB 2180                               [855]
2080 text$="":xp=xpo                          [1259]
2090 :                                        [174]
2100 GOSUB 2250                               [865]
2110 IF i=CHR$(13)GOTO 2140                   [1290]
2120 IF i>="0"AND i<="9"THEN GOSUB 2490       [2031]
2130 GOTO 2100                                [359]
2140 tval=VAL(text$)                          [1413]
2150 IF tval<tlo OR tval>thi GOTO 2080        [3204]
2160 PRINT:CLS#4                              [335]
2170 RETURN                                   [555]
2180 '                                        [117]
2190 '*** Help ***                            [971]
2200 '                                        [117]
2210 PRINT#4,"[DEL] links,[CLR] rechts,["C    [5720]
HR$(&F2)"] Cursor links,["CHR$(&F3)"] Curs
or rechts,[ENTER] beenden"
2220 RETURN                                   [555]
2230 PRINT#4,"["CHR$(&F0)"] Cursor nach ob    [8687]
en,["CHR$(&F1)"] Cursor nach unten,[DEL] a
endern,[ENTER] weiter"
2240 RETURN                                   [555]
2250 '                                        [117]
```



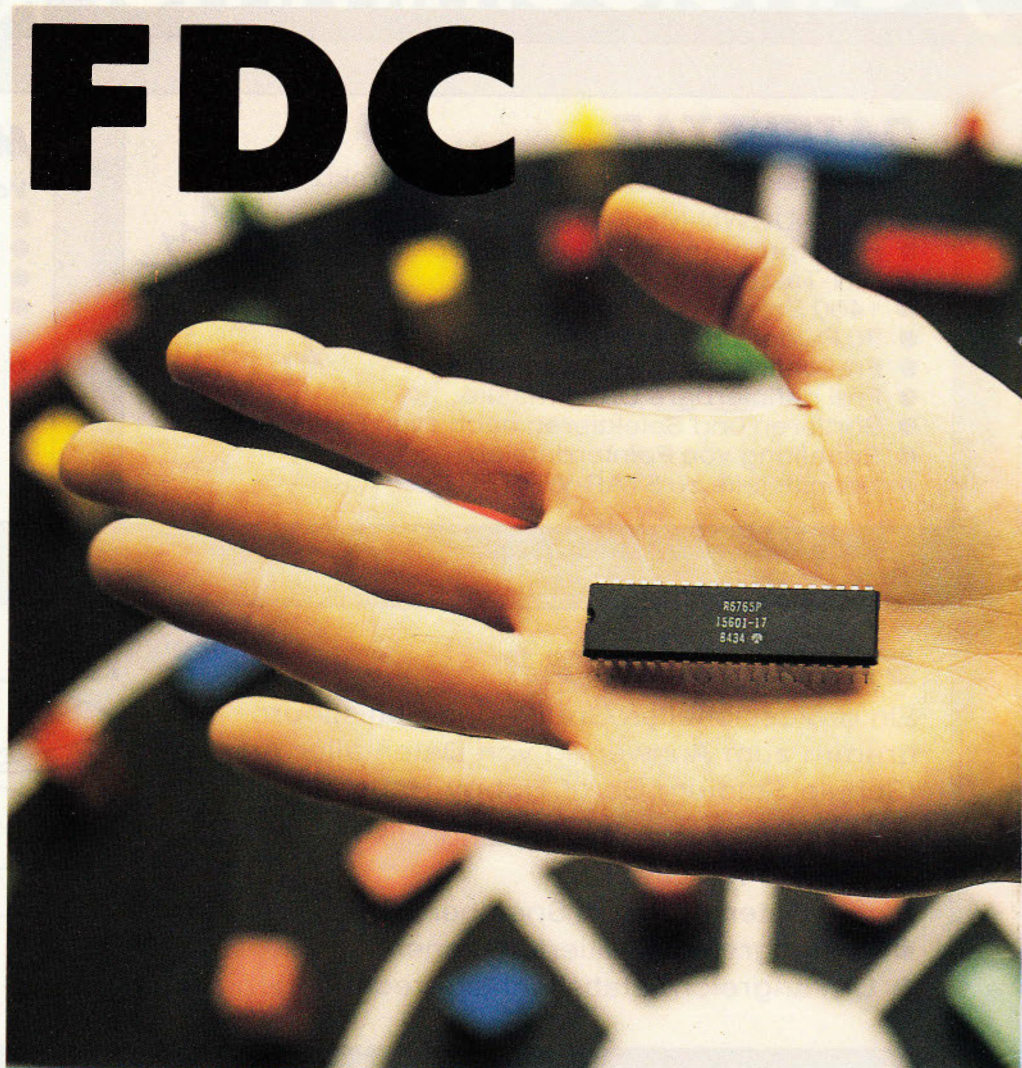
```
2260 '*** INPUT ***                           [671]
2270 '                                        [117]
2280 txt=LEN(text$)                           [1212]
2290 LOCATE xpo,yp                            [1408]
2300 PRINT text$+MID$(back$,txt+1)            [1964]
2310 LOCATE xp,yp                             [935]
2320 CALL &BB8A                               [558]
2330 i=INKEY$                                 [560]
2340 IF i=""GOTO 2330                         [785]
2350 nn=INSTR(bef1$,i)                        [1032]
2360 CALL &BB8D                               [555]
2370 ON nn GOTO 2390,2410,2430,2460           [1652]
2380 RETURN                                   [555]
2390 xp=MAX(xp-1,xpo)                         [1620]
2400 RETURN                                   [555]
2410 xp=MIN(xp+1,xpo+txt)                     [2367]
2420 RETURN                                   [555]
2430 x=xp-xpo                                 [376]
2440 IF x THEN text$=LEFT$(text$,x-1)+MID$    [3872]
(text$,x+1):xp=xp-1
2450 RETURN                                   [555]
2460 x=xp-xpo                                 [376]
2470 IF x<txt THEN text$=LEFT$(text$,x)+MI    [3685]
D$(text$,x+2)
2480 RETURN                                   [555]
2490 IF nn THEN RETURN                        [1564]
2500 x=xp-xpo                                 [376]
2510 IF x<t THEN text$=LEFT$(text$,x)+i+MI    [6201]
D$(text$,x+1):xp=xp+1
2520 RETURN                                   [555]
2530 '                                        [117]
2540 '*** Dateiname ***                       [1686]
2550 '                                        [117]
2560 PRINT"Dateiname ([ENTER] beenden): "C    [3352]
HR$(24);
2570 t=8:GOSUB 1840                           [1323]
2580 PRINT CHR$(24)                           [996]
2590 IF text$=""THEN RETURN                   [1128]
2600 file$=text$+".RDF"                       [1367]
2610 |FIND,@file$                             [1350]
2620 flag=PEEK(&960E)                         [974]
2630 RETURN                                   [555]
2640 '                                        [117]
2650 '*** ERROR ***                           [916]
2660 '                                        [117]
2670 ON er GOTO 2690,2710,2730,2750,2770      [1161]
2680 RETURN                                   [555]
2690 PRINT"<<< Datei besteht bereits >>>"     [3250]
2700 RETURN                                   [555]
2710 PRINT"<<< Datei ist noch geoeffnet >>    [4234]
>"
2720 RETURN                                   [555]
2730 PRINT"<<< Datei nicht gefunden >>>"      [3559]
2740 RETURN                                   [555]
2750 PRINT"<<< Datei ist geschlossen >>>"     [4361]
2760 RETURN                                   [555]
2770 PRINT"<<< Datei ist voll >>>"            [2862]
2780 RETURN                                   [555]
2790 '                                        [117]
2800 '*** RECORD ***                          [1043]
2810 '                                        [117]
2820 bytes=rec*reclen                         [1811]
2830 blk=INT(bytes/512)                       [2704]
2840 rst=bytes-512*blk                        [1408]
2850 |RECORD,blk,rst                          [863]
2860 RETURN                                   [555]
2870 '                                        [117]
2880 '*** GET 1 ***                           [965]
2890 '                                        [117]
2900 FOR f=0 TO felder-1                      [1536]
2910 |GET,f,@i                                [1354]
2920 mem1$(f)=i                               [840]
2930 NEXT:RETURN                              [940]
2940 '                                        [117]
2950 '*** PUT 1 ***                           [586]
2960 '                                        [117]
2970 FOR f=0 TO felder-1                      [1536]
2980 i=mem1$(f)                               [1329]
2990 |PUT,f,@i                                [627]
3000 NEXT:RETURN                              [940]
3010 '                                        [117]
3020 '*** GET 2 ***                           [813]
3030 '                                        [117]
3040 FOR f=0 TO felder-1                      [1536]
3050 |GET,f,@i                                [1354]
3060 mem2$(f)=i                               [631]
3070 NEXT:RETURN                              [940]
3080 '                                        [117]
3090 '*** PUT 2 ***                           [562]
3100 '                                        [117]
3110 FOR f=0 TO felder-1                      [1536]
3120 i=mem2$(f)                               [1377]
3130 |PUT,f,@i                                [627]
3140 NEXT:RETURN                              [940]
3150 '                                        [117]
3160 '*** MAX(1,2) ***                        [1230]
3170 '                                        [117]
3180 IF LOWER$(mem1$(fld))>LOWER$(mem2$(fl    [2446]
d))GOTO 3260
3190 RETURN                                   [555]
3200 '                                        [117]
3210 '*** MIN(1,2) ***                        [1232]
3220 '                                        [117]
3230 IF LOWER$(mem1$(fld))<LOWER$(mem2$(fl    [2387]
d))GOTO 3260
3240 RETURN                                   [555]
3250 :                                        [174]
3260 ex=m:rec=ex                              [948]
3270 GOSUB 2790                               [829]
3280 GOTO 2870                                [389]
3290 '                                        [117]
3300 '*** Maske ***                           [1142]
3310 '                                        [117]
3320 CLS                                      [91]
3330 FOR f=1 TO felder                        [1263]
3340 LOCATE 1,3+ftab(f-1)                     [1657]
3350 PRINT CHR$(24)" "fbez$(f)": "CHR$(24)    [1918]
3360 x1=flen(f)\56+1                          [1205]
3370 x2=((flen(f)MOD 56)>0)                   [1616]
3380 ftab(f)=ftab(f-1)+x1-x2                  [1983]
3390 NEXT:RETURN                              [940]
3400 '                                        [117]
```



```
3410 '*** Datensatz eingeben ***                [1457]
3420 '                                          [117]
3430 PRINT CHR$(30)"Satznummer:"rec             [2463]
3440 GOSUB 2790                                 [829]
3450 FOR f=0 TO felder-1                        [1536]
3460 LOCATE 1,ftab(f)+4                         [1398]
3470 t=flen(f+1):GOSUB 1840                     [1449]
3480 IF text$=""THEN RETURN                     [1128]
3490 mem1$(f)=text$                             [947]
3500 NEXT:RETURN                                [940]
3510 '                                          [117]
3520 '*** Datei errichten ***                   [2343]
3530 '                                          [117]
3540 IF PEEK(&9600)THEN er=2:GOTO 2640          [2101]
3550 GOSUB 2530                                 [865]
3560 IF text$=""THEN RETURN                     [1128]
3570 IF flag THEN er=1:GOTO 2640                [1615]
3580 name$=text$                                [1187]
3590 PRINT"Anzahl der Felder pro Datensatz      [3938]
 (max. 8): ";
3600 tlo=1:thi=8:GOSUB 2000                     [1842]
3610 PRINT                                      [361]
3620 PRINT"Nr./Laenge/Feldname"                 [1602]
3630 maxlen=512:felder=tval                     [2043]
3640 FOR n=1 TO felder                          [1167]
3650 thi=MIN(255,maxlen)                        [1795]
3660 PRINT n"("FNst$(thi)") ";                  [1150]
3670 GOSUB 2000                                 [875]
3680 LOCATE 4,yp                                [1006]
3690 PRINT USING"  ###     ";tval;              [2085]
3700 maxlen=maxlen-tval                         [2661]
3710 flen(n)=tval                               [2154]
3720 t=40:GOSUB 1840                            [1552]
3730 fbez$(n)=text$                             [1450]
3740 NEXT:PRINT                                 [811]
3750 |FIND                                      [992]
3760 bytes=PEEK(&960E)*1024-512                 [2047]
3770 reclen=512-maxlen                          [1466]
3780 maxrec=INT(bytes/reclen)                   [2566]
3790 belrec=0                                   [797]
3800 PRINT"Anzahl der Datensaetze (max. "F      [4662]
Nst$(maxrec)"): ";
3810 thi=maxrec:GOSUB 2000                      [1852]
3820 maxrec=tval                                [1790]
3830 bytes=maxrec*reclen                        [2052]
3840 blk=INT(bytes/255)                         [2789]
3850 rst=bytes-255*thi                          [1430]
3860 OPENOUT file$                              [1312]
3870 FOR n=1 TO 4                               [863]
3880 PRINT#9,header$;                           [615]
3890 NEXT                                       [350]
3900 FOR n=1 TO blk                             [931]
3910 PRINT#9,blank$;                            [1526]
3920 NEXT                                       [350]
3930 PRINT#9,LEFT$(blank$,rst);                 [1877]
3940 CLOSEOUT                                   [902]
3950 |OPEN,@file$                               [881]
3960 POKE &960A,FNlo(reclen)                    [913]
3970 POKE &960B,FNhi(reclen)                    [1169]
3980 POKE &9612,FNlo(maxrec)                    [1628]
3990 POKE &9613,FNhi(maxrec)                    [2330]
4000 FOR n=1 TO felder                          [1167]
4010 |ITEM,@fbez$(n),flen(n)                    [1576]
4020 NEXT                                       [350]
4030 '                                          [117]
4040 '*** Statuszeile ***                       [1463]
4050 '                                          [117]
4060 LOCATE#3,9,3                               [641]
4070 PRINT#3,UPPER$(name$)TAB(17);              [593]
4080 LOCATE#3,8,4                               [648]
4090 PRINT#3,belrec;TAB(14);                    [1205]
4100 LOCATE#3,8,5                               [651]
4110 PRINT#3,maxrec-belrec;TAB(14);             [2310]
4120 RETURN                                     [555]
4130 '                                          [117]
4140 '*** Datei loeschen ***                    [2470]
4150 '                                          [117]
4160 GOSUB 2530                                 [865]
4170 IF text$=""THEN RETURN                     [1128]
4180 IF flag=0 THEN er=3:GOTO 2640              [1802]
4190 |ERA,@file$                                [1422]
4200 RETURN                                     [555]
4210 '                                          [117]
4220 '*** Datei laden ***                       [2029]
4230 '                                          [117]
4240 IF PEEK(&9600)THEN er=2:GOTO 2640          [2101]
4250 GOSUB 2530                                 [865]
4260 IF text$=""THEN RETURN                     [1128]
4270 IF flag=0 THEN er=3:GOTO 2640              [1802]
4280 name$=text$                                [1187]
4290 |OPEN,@file$                               [881]
4300 felder=PEEK(&9605)                         [1899]
4310 PRINT"Dateistrucktur:"                     [2145]
4320 PRINT"Nr./Laenge/Feldname"                 [1602]
4330 p=&961A                                    [784]
4340 FOR n=1 TO felder                          [1167]
4350 |EMIT,@i                                   [1309]
4360 m=FNdp(p+2)-FNdp(p)                        [1584]
4370 PRINT n;USING"  ###     "+i;m              [1798]
4380 fbez$(n)=i                                 [1150]
4390 flen(n)=m                                  [1278]
4400 p=p+2:NEXT                                 [1394]
4410 maxrec=FNdp(&9612)                         [1405]
4420 belrec=FNdp(&9614)                         [1152]
4430 reclen=FNdp(&960A)                         [1843]
4440 GOTO 4030                                  [539]
4450 '                                          [117]
4460 '*** Datei sichern ***                     [1649]
4470 '                                          [117]
4480 IF PEEK(&9600)=0 THEN er=4:GOTO 2640       [2014]
4490 POKE &9614,FNlo(belrec)                    [1632]
4500 POKE &9615,FNhi(belrec)                    [1909]
4510 |CLOSE:RETURN                              [1120]
4520 '                                          [117]
4530 '*** Datei sortieren ***                   [2114]
4540 '                                          [117]
4550 IF PEEK(&9600)=0 THEN er=4:GOTO 2640       [2014]
4560 PRINT"[1] aufsteigende"                    [2486]
4570 PRINT"[2] absteigende Reihenfolge"         [3355]
4580 PRINT"[0] beenden"                         [1491]
4590 tlo=0:thi=2:GOSUB 2000                     [1050]
4600 IF tval=0 THEN RETURN                      [1458]
4610 sort=tval                                  [884]
4620 PRINT                                      [361]
4630 PRINT"Nach welchem Feld soll sortiert      [4378]
 werden ?"
4640 GOSUB 4850                                 [921]
4650 FOR n=0 TO belrec-2                        [1760]
4660 ex=n:rec=ex                                [1260]
4670 GOSUB 2790                                 [829]
4680 GOSUB 2870                                 [901]
4690 FOR m=n+1 TO belrec-1                      [1803]
4700 rec=m                                      [487]
4710 GOSUB 2790                                 [829]
4720 GOSUB 3010                                 [967]
4730 ON sort GOSUB 3150,3200                    [1245]
4740 NEXT                                       [350]
4750 rec=n                                      [490]
4760 GOSUB 2790                                 [829]
4770 GOSUB 3010                                 [967]
4780 GOSUB 2940                                 [895]
4790 rec=ex                                     [487]
4800 GOSUB 2790                                 [829]
4810 GOSUB 3080                                 [1136]
4820 NEXT                                       [350]
4830 RETURN                                     [555]
4840 :                                          [174]
4850 PRINT                                      [361]
4860 PRINT"Nr./Feldname"                        [855]
4870 FOR n=1 TO felder                          [1167]
4880 PRINT n" "fbez$(n)                         [1807]
4890 NEXT:PRINT                                 [811]
4900 tlo=1:thi=felder                           [2220]
4910 GOSUB 2000                                 [875]
4920 fld=tval-1                                 [890]
```

```
4930 RETURN                                    [555]
4940 '                                         [117]
4950 '*** Directory zeigen ***                 [1676]
4960 '                                         [117]
4970 WINDOW 24,53,2,21                         [1105]
4980 file$="*.RDF"                             [1419]
4990 |DIR,@file$                               [1295]
5000 WINDOW 24,79,2,21                         [1171]
5010 RETURN                                    [555]
5020 '                                         [117]
5030 '*** Daten eingeben ***                   [2042]
5040 '                                         [117]
5050 IF PEEK(&9600)=0 THEN er=4:GOTO 2640      [2014]
5060 PRINT"[1] Datensaetze anhaengen"          [1985]
5070 PRINT"[2] Datensaetze einfuegen"          [1892]
5080 PRINT"[3] Datensaetze aendern"            [2218]
5090 PRINT"[0] beenden"                        [1491]
5100 PRINT                                     [361]
5110 tlo=0:thi=4                               [1025]
5120 GOSUB 2000                                [875]
5130 ON tval GOTO 5160,5260,5570               [1958]
5140 RETURN                                    [555]
5150 :                                         [174]
5160 GOSUB 3290                                [1198]
5170 WHILE belrec<maxrec                       [2733]
5180 rec=belrec                                [829]
5190 GOSUB 3400                                [1009]
5200 IF text$=""GOTO 4030                      [841]
5210 GOSUB 2940                                [895]
5220 belrec=belrec+1                           [1172]
5230 WEND                                      [390]
5240 er=5:GOTO 2640                            [1672]
5250 :                                         [174]
5260 GOSUB 5460                                [871]
5270 IF frerec=0 THEN RETURN                   [2434]
5280 IF belrec+frerec>maxrec THEN er=5:GOT     [4096]
O 2640
5290 FOR n=1 TO belrec-posrec                  [2176]
5300 rec=belrec-n                              [948]
5310 GOSUB 2790                                [829]
5320 GOSUB 2870                                [901]
5330 rec=rec+frerec                            [1727]
5340 GOSUB 2790                                [829]
5350 GOSUB 2940                                [895]
5360 NEXT                                      [350]
5370 GOSUB 3290                                [1198]
5380 belrec=belrec+frerec                      [1655]
5390 FOR n=0 TO frerec-1                       [1801]
5400 rec=posrec+n                              [1720]
5410 GOSUB 3400                                [1009]
5420 IF text$=""GOTO 4030                      [841]
5430 GOSUB 2940                                [895]
5440 NEXT:GOTO 4030                            [969]
5450 :                                         [174]
5460 IF belrec=0 THEN frerec=0:RETURN          [920]
5470 PRINT"Satznummer:";                       [1705]
5480 tlo=0:thi=belrec-1                        [1396]
5490 GOSUB 2000                                [875]
5500 posrec=tval                               [721]
5510 PRINT"     Anzahl:";                      [1701]
5520 thi=belrec-posrec                         [1642]
5530 GOSUB 2000                                [875]
5540 frerec=tval                               [2290]
5550 RETURN                                    [555]
5560 :                                         [174]
5570 GOSUB 5460                                [871]
5580 GOSUB 3290                                [1198]
5590 FOR n=0 TO frerec-1                       [1801]
5600 rec=posrec+n                              [1720]
5610 GOSUB 2790                                [829]
5620 GOSUB 6350                                [853]
5630 GOSUB 5660                                [974]
5640 NEXT:RETURN                               [940]
5650 :                                         [174]
5660 GOSUB 2230                                [821]
5670 y=0                                       [483]
5680 LOCATE 1,ftab(y)+4                        [1264]
5690 CALL &BB8A                                [558]
5700 i=INKEY$                                  [560]
5710 IF i=""GOTO 5700                          [942]
5720 nn=INSTR(bef2$,i)                         [1427]
5730 CALL &BB8D                                [555]
5740 IF i=CHR$(13)THEN CLS#4:RETURN            [973]
5750 ON nn GOSUB 5770,5790,5810                [1087]
5760 GOTO 5680                                 [427]
5770 y=MAX(y-1,0)                              [767]
5780 RETURN                                    [555]
5790 y=MIN(y+1,felder-1)                       [1930]
5800 RETURN                                    [555]
5810 t=flen(y+1):GOSUB 1840                    [1936]
5820 IF text$=""THEN RETURN                    [1128]
5830 mem1$(y)=text$                            [1380]
5840 GOTO 2940                                 [383]
5850 '                                         [117]
5860 '*** Daten auflisten ***                  [1519]
5870 '                                         [117]
5880 IF PEEK(&9600)=0 THEN er=4:GOTO 2640      [2014]
5890 PRINT"[1] Alles auflisten"                [2552]
5900 PRINT"[2] bedingte Ausgabe"               [1819]
5910 PRINT"[0] beenden":PRINT                  [2206]
5920 tlo=0:thi=2                               [1011]
5930 GOSUB 2000                                [875]
5940 ON tval GOTO 5970,6050                    [1339]
5950 RETURN                                    [555]
5960 :                                         [174]
5970 GOSUB 3290                                [1198]
5980 FOR rec=0 TO belrec-1                     [1405]
5990 GOSUB 2790                                [829]
6000 GOSUB 6350                                [853]
6010 CALL &BB06                                [393]
6020 NEXT                                      [350]
6030 RETURN                                    [555]
6040 :                                         [174]
6050 PRINT                                     [361]
6060 PRINT"Nach welchem Feld soll gesucht      [3536]
werden ?"
6070 GOSUB 4850                                [921]
6080 PRINT"Suchmaske: ";                       [2172]
6090 t=flen(fld):GOSUB 1840                    [1984]
6100 IF text$=""THEN RETURN                    [1128]
6110 mask$=LOWER$(text$)                       [2532]
6120 GOSUB 3290                                [1198]
6130 FOR rec=0 TO belrec-1                     [1405]
6140 GOSUB 2790                                [829]
6150 GOSUB 2870                                [901]
6160 text$=LOWER$(mem1$(fld))                  [2067]
6170 GOSUB 6200                                [899]
6180 NEXT:RETURN                               [940]
6190 :                                         [174]
6200 m=1                                       [561]
6210 FOR n=1 TO LEN(mask$)                     [2243]
6220 bst$=MID$(mask$,n,1)                      [2283]
6230 IF bst$="*"GOTO 6320                      [1673]
6240 IF bst$="?"GOTO 6260                      [1600]
6250 IF bst$<>MID$(text$,m,1)THEN RETURN       [3283]
6260 m=m+1                                     [1053]
6270 NEXT                                      [350]
6280 GOSUB 6390                                [877]
6290 CALL &BB06                                [393]
6300 RETURN                                    [555]
6310 :                                         [174]
6320 m=INSTR(m,text$,CHR$(44))                 [3326]
6330 IF m=0 GOTO 6280                          [1489]
6340 GOTO 6270                                 [397]
6350 '                                         [117]
6360 '*** PRINT ***                            [1131]
6370 '                                         [117]
6380 GOSUB 2870                                [901]
6390 PRINT CHR$(30)"Satznummer:"rec            [2463]
6400 FOR f=0 TO felder-1                       [1536]
6410 LOCATE 1,4+ftab(f)                        [1572]
6420 PRINT mem1$(f)                            [594]
6430 NEXT:RETURN                               [940]
6440 '                                         [117]
6450 '*** Daten loeschen ***                   [1826]
6460 '                                         [117]
6470 IF PEEK(&9600)=0 THEN er=4:GOTO 2640      [2014]
6480 GOSUB 5460                                [871]
6490 IF frerec=0 THEN RETURN                   [2434]
6500 FOR n=0 TO belrec-posrec-1                [2090]
6510 rec=posrec+n+frerec                       [1992]
6520 GOSUB 2790                                [829]
6530 GOSUB 2870                                [901]
6540 rec=posrec+n                              [1720]
6550 GOSUB 2790                                [829]
6560 GOSUB 2940                                [895]
6570 NEXT                                      [350]
6580 belrec=belrec-frerec                      [2393]
6590 GOTO 4030                                 [539]
6600 '                                         [117]
6610 '*** Zurueck zu BASIC ***                 [1852]
6620 '                                         [117]
6630 INPUT"Wirklich (j/n)";i                   [1927]
6640 IF LOWER$(i)<>"j"GOTO 1260                [881]
6650 GOSUB 4490                                [1023]
6660 MODE 1                                    [506]
6670 PRINT"Neustart mit 'GOTO 1030'"           [3424]
6680 CLEAR                                     [217]
```

Der universelle Floppy-Disk-Controller µPD 765 im Schneider-Floppylaufwerk



Der 765 ist ein eigenständiger Mikroprozessor, der speziell für die Verwaltung der Daten auf der Diskette, den Datenaustausch mit dem Rechner und die Steuerung der Laufwerksmechanik der Floppystation konzipiert ist. Von Haus aus ist der 765 sehr leistungsfähig; er ermöglicht z.B. den Datentransfer mit einem DMA-Baustein, ohne den Rechnerprozessor zu belästigen. Ebenso ist die Verwaltung von bis zu vier Laufwerken oder die Ansteuerung von Doppelkopflaufwerken möglich. Diese Optionen werden allerdings vom Diskverwaltungsprogramm AMSDOS nicht genutzt und sollen hier auch nicht weiter erläutert werden. Dieser Bericht soll einen Einblick in die Arbeit des FDC und in die Zusammenarbeit mit der Z80A geben; Bild 1 stellt diese Zusammenarbeit schematisch dar.

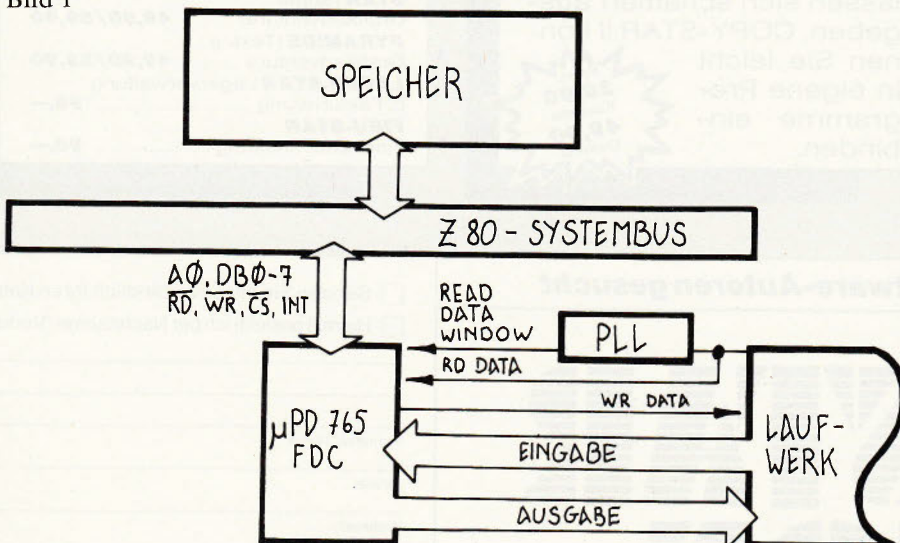
Die Pinbelegung

Bild 2 stellt die Anschlußbelegung des µPD 765 dar; die Funktionen der einzelnen Pins können Sie der folgenden Aufstellung entnehmen, wobei eine Erläuterung der von AMSDOS nicht genutzten Funktionen nicht vorgenommen wird.

Pin 1: RESET
Durch Anlegen einer logischen '1' = 'High'-Potential wird der 765 in den Einschaltzustand versetzt.

Pin 2: ReaD
Ist verbunden mit dem RD-pin der Z80A, wird von dieser auf logisch '0'='low'-Potential gelegt, wenn Daten vom 765 gelesen werden sollen.

Bild 1



Pin 3: WRite
Wie Pin 2, wird vom Prozessor low gesetzt, wenn Daten in den FDC geschrieben werden sollen.

Pin 4: Chip Select
Wird vom Prozessor bedient. Wenn low, wird der 765 als Partner der CPU aktiviert, RD und WR werden ermöglicht.

Pin 5: Adress line 0
Wird vom Prozessor bedient. Wenn high, wird der Inhalt des Datenregisters an den Datenbus gelegt, low legt den Inhalt des Statusregisters an.

Pins 6 - 13: Data Bus 0 - 7
Über den acht Bit breiten Datenbus empfängt der 765 Daten und Befehle vom Prozessor, sendet Daten und den Inhalt des Statusregisters.

Pin 14: Data dma ReQuest
Wird benutzt im DMA-Modus, d.h. in der Datenübertragung ohne Prozessor.

Pin 15: Dma ACKnowledge
wie Pin 14

Pin 16: Terminal Count
wie Pin 14

Pin 17: InDeX
Erhält High-Signal vom Floppylaufwerk, wenn die Lichtschranke das Indexloch erkannt hat, das bedeutet: Anfang der Disk-Spur.

Pin 18: INTerrupt
Wird vom FDC high gesetzt, um die Aufmerksamkeit des Prozessors z.B. bei der Datenübertragung zu erlangen.

Pin 19: ClocK



Der Takteingang; muß extern mit einem 4/8 MHz Rechtecksignal gespeist werden (Schneider: 4 MHz).

Pin 20: GrouND
Systemmasse, gleichzeitig Anschluß der negativen Versorgungsspannung.

Pin 21: Write ClocK
An diesen Pin wird ein Rechtecksignal gelegt, (0,5/1 MHz), welches die Übertragungsgeschwindigkeit zu senden-der/zu empfangender Daten bestimmt.

Pin 22: Read Data Window
Dieses Signal wird vom PLL-Generator erzeugt und dient dem Zusammensetzen eines Bytes aus den vom Laufwerk gelesenen Daten.

Pin 23: Read Data from Disk
Über diesen Anschluß werden seriell die auf Disk gespeicherten Informationen einschließlich des Synchronisationssignals eingelesen.

Pin 24: Voltage Controlled Oscillator
Schaltet den VCO ein, wenn high.

Pin 25: Write Enable
Wird u.a. im Zusammenhang mit geschützten Disketten benutzt: Wenn low, ist kein Schreibzugriff auf Diskette möglich.

Pin 26: MFM Mode
Dies ist der Umschalter zwischen Single/Double Density-Format, wobei high doppelte Schreibdichte ist.

Pin 27: HeaD select
Wählt die Köpfe eines Doppellaufwerks an: Wenn high, ist Kopf 1 gewählt.

Pins 28,29: Unit Select 1 + 2
Mit diesen Pins wird das gerade vom 765 angesprochene Laufwerk aktiviert.

Pin 30: Write DAta
An diesen Pin werden die vom Laufwerk gelesenen Daten und das Synchronisationssignal angelegt.

Pins 31,32: Pre-Shift 1 + 2
Mit dem an diesen Pins anliegenden Signal werden Zeitunterschiede im Double-Density-Modus kompensiert. Mögliche Ausgaben sind 'früh, normal und spät'.

Pin 33: FauLT/TRack0
Wenn high im Read/Write-Modus, dann ist ein Laufwerksfehler aufgetreten, wenn high im Seek-Modus, dann ist Track 0 erreicht.

Pin 34: Write Protect/Two-Side
Wird vom Laufwerk im Read/Write-Modus high gesetzt, wenn die Diskette schreibgeschützt ist, im Seek-Modus wird High-Potential angelegt, wenn eine doppelseitig beschreibbare Diskette erkannt wurde.

Pin 35: ReaDY
Dieser Anschluß wird vom Laufwerk high gesetzt, wenn dieses zum Senden oder Empfangen von Daten bereit ist, nachdem der Laufwerksmotor seine Arbeitsgeschwindigkeit erreicht hat.

Pin 36: HeaD Load
Für Laufwerke (hauptsächlich 8"-Formate), deren Laufwerksmotor ständig eingeschaltet ist, um den Schreib-/Lesekopf bei Bedarf mechanisch auf die Diskette abzusenken oder von ihr abzuheben.

Pin 37: Fit Reset/STeP
Im Read/Write-Modus wird durch anliegendes High-Potential ein ggfs. in einem Laufwerk befindliches Fehler-Flip-Flop zurückgesetzt (Fehler ist beseitigt), im Seek-Modus ist ein High-Impuls die Anweisung an den Steppermotor, einen Schritt in die durch LCT/DIR (Pin 38) angegebene Richtung zu machen.

Pin 38: Low CurrenT/DIRection
Wird im Read/Write-Modus dieser Pin high gesetzt, so wird von der Steuer-Elektronik des Laufwerks, der Schreibstrom zum Beschreiben der inneren Spuren vermindert; im Seek-Modus wird durch Anlegen von high oder low die Richtung der Steppermotorbewegung angegeben.

Pin 39: Read/Write-Seek
An diesem Pin wird nun der Arbeitsmodus des Laufwerks festgelegt. Lese-/Schreibzugriffe werden mit low angewählt, durch Anlegen von High-Potential wird der Seek-Modus eingeschaltet, in dem das Positionieren des Schreib-/Lesekopfes erfolgt.

Pin 40: VCC
Hier wird die Versorgungsspannung (+5V +/- 5 %) angelegt.

Die Befehlsübermittlung

Der µPD 765 erhält seine Kommandos vom Prozessor des Hauptsystems (in diesem Fall von der Z80A-CPU) in Form mehrerer acht Bit breiter Kommandoworte, die die auszuführende Aktion detailliert beschreiben und sämtliche Parameter enthalten. Diese Worte werden im Handshakebetrieb auf den Datenbus des 765 gelegt. Das Abarbeiten eines Befehls geschieht in drei Phasen:

1. Kommandophase
Der 765 erhält die Kommandoworte, die zur Ausführung eines Befehls benötigt werden und quittiert deren Empfang.

2. Ausführungsphase
Der übermittelte Befehl wird ausgeführt.

3. Ergebnisphase
Nach der vollständigen Ausführung stellt der 765 den Status der Operation und weitere Informationen wie z.B. die Nummer der aktuellen Spur und des Sektors an seinem Datenbus zur Verfügung.

Nachfolgend wird der Befehlssatz des 765 aufgelistet und beschrieben. Jeder Befehl wird in den beschriebenen Phasen erklärt, die Kommandoworte werden in Bits aufgelöst beschrieben, wobei ein gesetztes Bit mit '1', ein gelöschtes mit '0' gekennzeichnet ist. Die Bedeutung der Symbole entnehmen Sie bitte der Tabelle 1. Links im Kommandowort steht das höchstwertige Bit, welches am Datenbus D7 angelegt wird. Die in Phase 1 unter 1. und 2. angegebenen Kommandoworte spezifizieren den Befehl, die unter 3. aufgeführten Sektor-ID-Informationen enthalten ebenfalls ein Datenbyte, welches z.B. den zu bearbeitenden Sektor angibt. Die Kommando- und Informationsworte werden nacheinander vom Hauptprozessor auf den Datenbus des 765 gelegt, um die gewünschte Aktion genau zu beschreiben.

1. READ DATA
Phase 1:
1. MT MF SK 0 0 1 1 0
2. 0 0 0 0 0 HD US1 US1
3. Sektor-ID-Information (in dieser Reihenfolge): C,H,R,N,EOT,GPL,DTL
Phase 2: Ausführung, Datentransfer zwischen Floppy und Prozessor
Phase 3: Status- und Sektorinformatio-

nen (in dieser Reihenfolge): ST0,ST1, ST2,C,H,R,N
Ablauf: Positionieren, Suchen der angegebenen Sektor-ID, Datenübertragung, Ergebnisse mitteilen.

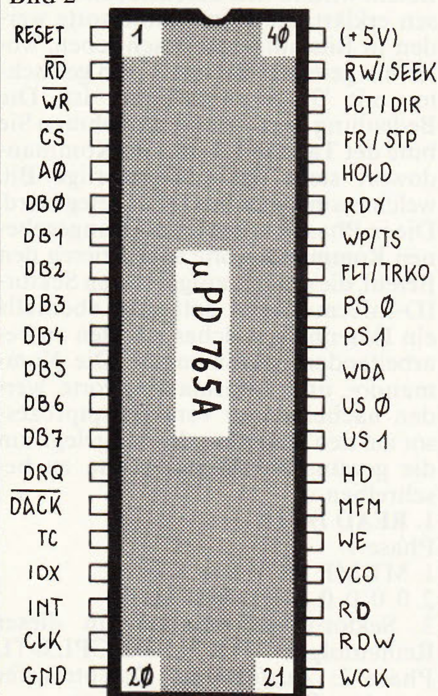
2. READ DELETED DATA
Phase 1:
1. MT MF SK 0 1 1 0 0
2. 0 0 0 0 0 HD US1 US0
3. Sektor-ID-Information: C,H,R,N,EOT,GPL,DTL
Phase 2: Datentransfer
Phase 3: Status- und Sektorinformationen: ST0,ST1,ST2,C,H,R,N
Ablauf: Positionieren, Suchen der Sektor-ID, Überspringen der Markierung für gelöschte Sektoren, Datenübertragung, Ergebnisse.

3. WRITE DATA
Phase 1:
1. MT MF 0 0 0 1 0 1
2. 0 0 0 0 0 HD US1 US0
3. Wie READ DATA
Phase 2: Wie READ DATA
Phase 3: Wie READ DATA
Ablauf: Wie READ DATA

4. WRITE DELETED DATA
Phase 1:
1. MT MF 0 0 1 0 0 1
2. 0 0 0 0 0 HD US1 US0
3. Wie READ DELETED DATA
Phase 2: Wie READ DELETED DATA
Phase 3: Wie READ DELETED DATA
Ablauf: Wie READ DELETED DATA

5. READ A TRACK
Phase 1:
1. 0 MF SK 0 0 0 1 0
2. 0 0 0 0 0 HD US1 US0
3. Sektor-ID-Informationen übermitteln: C,H,R,N,EOT,GPL,DT
Phase 2: Datentransfer alle Daten vom Indexloch bis zur EOT-Markierung werden gelesen.
Phase 3: Rückgabe der Status- und -ID-Informationen: ST0,ST1,ST2, H,R,N
Ablauf: Positionieren auf den gewünschten Track, Start des Lesevorgangs nach Erkennen des Indexloches, Ende beim gesetzten EOT oder nachdem das Indexloch ein zweites Mal erkannt wurde.

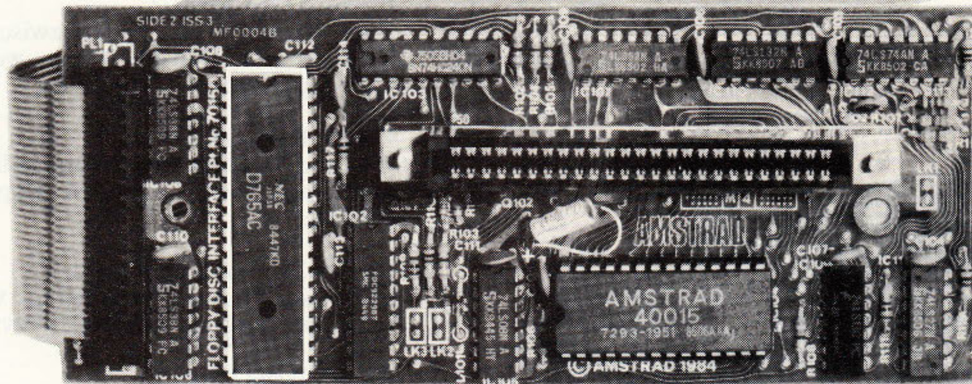
Bild 2



6. READ ID
Phase 1:
1. 0 MF 0 0 1 0 1 0
2. 0 0 0 0 0 HD US1 US2
Phase 2:
Die erste gültige ID auf einem Track wird gelesen und in das Datenregister übergeben.
Phase 3: Übergabe der Sektor-ID und Statusinformationen: ST0,ST1,ST2,C, H,R,N
Ablauf: Positionieren auf den gewünschten Track, Lesen der ersten gültigen ID des Tracks und Übergabe der ID an den Datenport, Ergebnisse (Status) der Operation mitteilen.

7. FORMAT A TRACK
Phase 1:
1. 0 MF 0 0 1 1 0 1
2. 0 0 0 0 0 HD US1 US0
3. Angeben, welcher Track formatiert werden soll, wieviele Bytes pro Sektor zu füllen sind und welchen Inhalt das Füllerbyte hat: N,SC,GPL,D
Phase 2: Laufwerk formatiert einen bestimmten Track.
Phase 3: Nur die Statusinformation wird ausgegeben. Bytefolge: ST0,ST1, ST2,C,H,R,N
Ablauf: Positionieren auf den zu formatierenden Track, nach Erkennen des Indexlochs werden die ID's, die Adressmarkierungen und die Füllbytes auf die vorher festgelegte Anzahl von Sektoren geschrieben, anschließend wird der Status ausgegeben.

8. SCAN EQUAL
Phase 1:
1. MT MF SK 1 0 0 0 1
2. 0 0 0 0 0 HD US1 US0
3. Sektor-ID-Informationen: C,H,R,N, EOT,GPL,STP
Phase 2: Vergleich der im angegebenen Sektor geschriebenen Daten mit den zu schreibenden zwischen Laufwerk und Hauptprozessor.
Phase 3: Übergabe der Status- und Sektor-ID-Informationen: ST0,ST1, ST2,C,H,R,N
Ablauf: Lesen der im angegebenen



Sektor enthaltenen Daten, Anfordern eines Bytes der 'Originaldaten' vom Hauptprozessor, Vergleichen dieses Bytes nach der Bedingung 'gleich', Ende des Vergleichs, wenn die Testbedingung erfüllt wurde, Rückgabe der Statusinformation. Dieser Befehl erlaubt eine komfortable 'Verify'-Funktion.

9. SCAN LOW OR EQUAL
Phase 1:
1. MT MF SK 1 1 0 0 1
2. 0 0 0 0 0 HD US1 US0
3. Wie SCAN EQUAL
Phase 2: Wie SCAN EQUAL
Phase 3: Wie SCAN EQUAL
Ablauf: Wie SCAN EQUAL, allerdings werden die Bytes bei diesem Befehl nach der Bedingung 'kleiner oder gleich' verglichen.

10. SCAN HIGH OR EQUAL
Phase 1:
1. MT MF SK 1 1 1 01
2. 0 0 0 0 0 HD US1 US0
3. Wie SCAN EQUAL
Phase 2: Wie SCAN EQUAL
Phase 3: Wie SCAN EQUAL
Ablauf: Wie SCAN EQUAL, allerdings werden die Bytes bei diesem Befehl nach der Bedingung 'größer oder gleich' verglichen.

11. RECALIBRATE
Phase 1:
1. 0 0 0 0 0 1 1 1
2. 0 0 0 0 0 0 US1 US0
Phase 2: Der Schreib-/Lesekopf des Laufwerks wird auf Track 0 zurückgesetzt.
Ausführung: Die 765-internen Flags für die aktuelle Kopfposition werden auf null gesetzt; der Kopf wird so lange in Richtung des Tracks 0 geführt, bis das Laufwerk den 'Track0'-Impuls ausgibt.

Der Status dieser Operation kann in den Statusregistern abgefragt werden.

12. SENSE INTERRUPT STATUS
Phase 1:
1: 0 0 0 0 0 0 0 0
Phase 3: Übergabe des Status des 765 nach einem Interrupt am Ende einer SEEK- oder RECALIBRATE-Befehlsfolge durch ST0,PCN

13. SPECIFY
Phase 1:
1. 0 0 0 0 0 0 1 1

3. Übergabe der Standardwerte für die Initialisierung eines Laufwerks: SRT,HUT,HLT,ND

14. SENSE DRIVE STATUS
Phase 1:
1. 0 0 0 0 0 1 0 0
2. 0 0 0 0 0 HD US1 US0
Phase 3:
Übergabe des Laufwerkstatus an den Hauptprozessor durch Statusregister 3: ST3

15. SEEK
Phase 1:
1. 0 0 0 0 1 1 1 1
2. 0 0 0 0 0 HD US1 US2
3. Angabe der neuen Track-Nummer: NCN
Phase 2: Durch Ausgabe von Impulsen an die Steuerelektronik des Laufwerks wird der Schreib-/Lesekopf auf die gewünschte Position gebracht.

Die Statusregister

Der FDC 765 ist mit fünf Registern ausgestattet, die dem Prozessor des Hauptsystems ermöglichen, (fast) jederzeit über den Status des Laufwerks des FDC und ausgeführter Operationen informiert zu sein. Das Hauptstatusregister zeigt den Zustand des FDC an und übernimmt den Floppy-seitigen Quittungs-(Handshake)-Betrieb. Es kann jederzeit ausgelesen werden. Die Register ST0 – ST2 stellen Meldungen über den erfolgreichen Abschluß von Lese- und Schreiboperationen zur Verfügung. Das Auslesen ist nur bei ST0 über eigene Befehle möglich. In ST3 schließlich stehen die kompletten Statusinformationen des Laufwerks zur Verfügung und können vom Anwender über geeignete Befehle abgerufen werden. Die Belegung des Hauptregisters und des Laufwerksregisters ST3 und deren mögliche Inhalte sind nachfolgend aufgelistet.

Das Hauptstatusregister

Bit 7: ReQuest for Master
Wenn high, ist der 765 zum Datentransfer mit dem Hauptprozessor bereit.

Bit 6: Data Input/Output
High = senden, low = empfangen

Bit 5: EXecution Mode
High, wenn die Ausführungsphase begonnen hat, low vor Beginn der Ergebnisphase.

Bit 4: fdC Busy
Wenn der 765 während einer Schreib-/Leseoperation keine weiteren Kommandos entgegennehmen kann, wird dieses Bit high gesetzt.

Bits 0 – 3: fdD3-0 Busy
Während einer SEEK- oder RECALIBRATE-Operation eines der vier möglichen Laufwerke wird das zugehörige Bit gesetzt, um dem Hauptprozessor zu signalisieren, daß zur Zeit kein Datentransfer stattfinden kann.

Das Statusregister 3

Bit 7: FaulT
Wird high, wenn ein beliebiges Fehlersignal vom Laufwerk kommt.

Bit 6: Write Protected
Wenn die eingelegte Diskette schreibgeschützt ist, wird dieses Bit high.

Bit 5: ReadY
High, wenn das Laufwerk zum Durchführen weiterer Operationen bereit ist.

Bit 4: Track 0
High, wenn der Schreib-/Lesekopf des Laufwerks in der Grundposition ist.

Bit 3: Two Side
Dieses Bit zeigt, welcher Laufwerkstyp (Einfach-/Doppellaufwerk) an den 765 angeschlossen ist (high = Einzellaufwerk).

Bit 2: HeaD address
Dieses Bit zeigt auf das Potential des an Pin 27 des 765 anliegenden Signals.

Bits 1,0: Unit Select
Diese Bits haben das gleiche Potential wie die Unit Select-Pins des 765.

Symbol	Name	Funktion
A0	Adress-Line 0	wählt Hauptstatusregister (0) oder Datenregister (1)
C	Cylinder-Number	Nummer des gewählten/aktuellen Tracks
D	Data	beschreibt Datenmuster, welches in einem Sektor geschrieben wird
D7 – D0	Data-Bus	8-Bit Datenbus; Bit 7 = High-Bit
DTL	Data-Length	wenn N = 0, wird hier die Länge der zu schreibenden/lesenden Daten übergeben
EOT	End of Track	bezeichnet den letzten Sektor einer Spur
GPL	Gap Length	enthält die Angabe, wie lang der VCO nach zwei Kommandobytes ausgeschaltet bleibt
H	Head-Adress	Kopfnummer in Doppelkopflaufwerken
HD	Head	= H in Kommandoworten, kontrolliert zusätzlich das Potential an Pin 27
HLT	Head load Time	2 – 254 ms in 2 ms-Stufen
HUT	Head unload Time	16 – 240 ms in 16 ms-Stufen
MF	FM/MFM-Mode	Umschaltung Single/Double-Density
MT	Multi-Track	zeigt die bevorstehende Ausführung einer Multi-Track-Operation an
N	Number	Anzahl der Bytes, die in einem Sektor geschrieben werden
NCN	New-Cylinder-Number	gewünschter Track, der nach dem Positionieren erreicht werden soll
ND	Non-DMA-Mode	wenn gesetzt, arbeitet der 765 mit dem Hauptprozessor zusammen
PCN	Present-Cylinder-Number	aktuelle Track-Nummer nach dem Positionieren
R	Record	Nummer des Sektors, auf den zugegriffen wird
R/W	Read-Write	Auswahl der Operation lesen/schreiben
SC	Sectors	gibt die Anzahl der Sektoren pro Track an
SK	Skip	Überspringen der Markierung 'gelöschte Daten'
SRT	Step-Rate-Time	Stepping-Zeit (1 – 16 ms in 1 ms-Schriften)
ST0 – 3	Status 0 – 3	diese Register stellen in der Ergebnisphase eines Befehls relative Informationen zur Verfügung
STP		während des SCAN-Befehls Auswahl der Sektoren
US0, US1	Unit-Select	Auswahl der Laufwerke 1 – 4

Technische Daten

Speisespannung 5V, max. 7V
Input low, min. -0.5V, max. 0.8V
Input high, min. 2V, max. 5.5V
Output low, max. 0.45V
Output high, min. 2.4V, max. 5V
Strombedarf 150 mA
Leistungsaufnahme 1W
Temperatur min. -10, max. +70 Grad

Die Daten über den FDC 765 wurden dem Datenblatt der Firma NEC entnommen. Sie sollen dem an Hardware interessierten Leser einen Überblick vermitteln und sind deshalb nicht in der nötigen Ausführlichkeit wiedergegeben. Lesern, die mit dem 765 eigene Hardware entwickeln möchten, gebe ich auf schriftliche Anfrage gern weitere Informationen zu diesem Baustein.

(ME)

Diskette – was ist das?

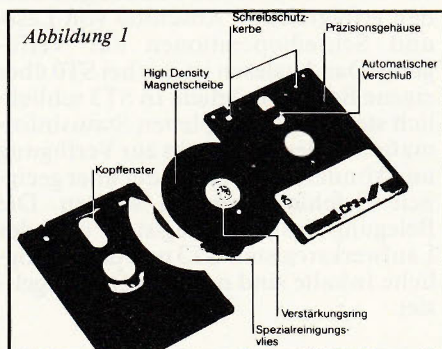
Neben dem herkömmlichen Datenträger Kassette, setzt sich zunehmend die Diskette als Speichermedium für Computerdaten durch. Die Diskettenspeicherung ist nicht nur wesentlich schneller, sondern auch um ein Vielfaches zuverlässiger als die Speicherung auf Kassette.

Die Diskette ist eine kleine Magnetscheibe aus flexiblem Material. In der Umgangssprache wird eine Diskette auch „Floppy-Disk" genannt. Man unterscheidet Disketten nach Größe, Aufzeichnungsdichte und Anzahl der verfügbaren Seiten zur Datenspeicherung.
Für den Schneider CPC sind nur die 3"- und 5 1/4"-Formate zu verwenden. Eine hohe, komplexe Fertigungstechnologie ist für die genaue Einhaltung der Formate unerläßlich, da die geringste Abweichung den Datenträger unbrauchbar macht.

Betrachten wir uns den Aufbau einer Diskette einmal am Beispiel der 3"-Compact-Floppy von Maxell (Abb. 1).
Die Compact-Floppy ist eine sogenannte High-Density Diskette, d.h., es wird durch eine Erhöhung der Speicherdichte die Aufzeichnungskapazität gesteigert. Voraussetzung für diese hohe Speicherdichte ist eine kompakte Magnetschicht – die Epitaxial-Beschichtung. Dieses Verfahren basiert auf Magnetpartikeln, die aus einer Verbindung von Kobalt-Ferrit und Gamma-Eisenoxid bestehen und mit einem Spezialbinder auf die Folie aufgetragen werden. Diese Beschichtung hat eine wesentlich höhere magnetische Kapazität als die üblichen Eisenoxid-Beschichtungen.
Das Kopffenster ist eine längliche Öffnung in der Hülle der Diskette, durch die der Schreib-/Lesekopf der Diskettenstation die Magnetschicht der Diskette magnetisiert (beim Schreiben) bzw. abtastet (beim Lesen).
An der Oberseite der Diskette sind sogenannte Schreibschutzkerben angebracht. Sie dienen dem Schutz der Daten gegen unbeabsichtigtes Überschreiben bzw. Löschen der gespeicherten Daten. Zwischen den Schreibschutzkerben befindet sich der automatische Verschluß, der die Compact-Floppy-Disk vor Schmutz und Staub schützt.
In der Mitte der Diskette sitzt das Center Hole, über das die Diskette vom Laufwerk angetrieben wird. Hier ist eine präzise Zentrierung für einen einwandfreien Lauf unbedingt Voraussetzung. Das nächste äußerliche Merkmal der 3"-Compact-Floppy ist das Indexloch. Dieses gestanzte Loch legt den physischen Spurenanfang der Diskette fest. Eine Lichtschranke in der Diskettenstation überwacht diese Stelle der Diskette und gibt jedesmal einen Impuls ab, wenn in einer Umdrehung der Strahlengang durch das Indexloch freigegeben wird.

Vorsicht! Diskette!

Abbildung 1



Fehlt zur Erläuterung noch das Reinigungsvlies, das zwischen Hülle und Magnetscheibe sitzt. Das antistatische Spezial-Reinigungsvlies reinigt permanent die Diskette und vermindert die Reibung zwischen Diskette und Hülle.
Das waren die wesentlichen Merkmale der 3"-Compact-Floppy von Maxell. Als Trägermaterial der Diskette dient eine Polyesterfolie von einem zehntel Millimeter Stärke. Auf die Folie wird beidseitig eine Schicht aufgebracht, in die die Epitaxial-Beschichtung als eigentlicher magnetischer Datenspeicher eingebettet ist. Diese Beschichtung wird getrocknet und geglättet. Danach werden die Scheiben ausgestanzt, wobei es besonders auf die Maßgenauigkeit des Center Hole ankommt. Die Diskette wird oberflächenversiegelt und poliert, die Hülle gefertigt und die Diskette darin versiegelt. Es folgsdie Prüfung auf „Herz und Nieren", die bei Maxell insgesamt 114 Einzelprüfungen ausmacht. Der sogenannte „Error-frei-Test" gilt als wichtigster Maßstab für die Qualität einer Diskette. Da die Spuren auf der Diskette nur Bruchteile von Millimetern breit sind und die Aufzeichnung mit großer Geschwindigkeit erfolgt, stellen umfangreiche Tests die Fehlerfreiheit aller verwendeten Materialien und Arbeitsgänge sicher.
Bekanntlich werden Daten auf einer Diskette in Blöcken gespeichert. Am Anfang und Ende jedes Blocks werden Lücken (GAP's) für Spuren- und Sektornummern sowie Prüfbytes gelassen. Diese GAP's grenzen einzelne Sätze einer Datei ab. Das Ende eines Satzes wird durch eine kleine Lücke (Unterbrechung) gekennzeichnet. Dieser Vorgang ist das Formatieren. Disketten sind durch den Hersteller formatiert (hard-sektoriert) oder unformatiert (soft-sektoriert) erhältlich.
Wir wissen nun, daß Disketten technisch hochwertige und zuverlässige Datenträger sind. Dementsprechend benötigen die Floppy-Disks auch Pflege und Gebrauchsvorschriften, um sie vor unbeabsichtigten Datenverlusten zu schützen.
Damit die Zuverlässigkeit und Funktionsfähigkeit der Disketten voll erhalten bleibt, sind folgende Punkte zu beachten:
Wenn Sie die hier gezeigten Bedienungshinweise genauestens beachten, sind Sie vor unfreiwilligen Datenverlusten geschützt.
Wir hoffen, daß Sie einen Einblick in die Funktions- und Bedienungshinweise der Diskette erhalten haben und würden uns freuen, wenn Ihnen beim Disk-Betrieb kein „Read-" oder „Write"-Error" unterkommt. (SR)

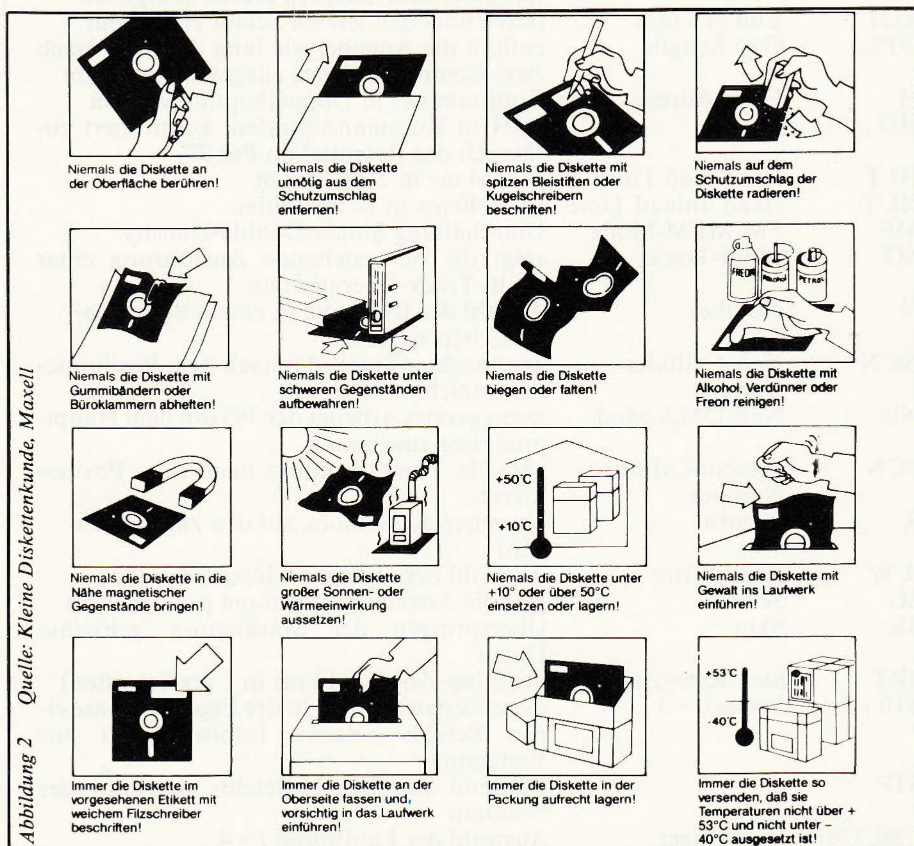


Abbildung 2 Quelle: Kleine Diskettenkunde, Maxell

AMSDOS INTERN

Das Disc-Basic in diesem Sonderheft ermöglicht Ihnen, den Aufbau Ihrer Disketten genau zu analysieren und Bereiche dieses Peripheriegerätes zu nutzen, die bisher noch im Verborgenen lagen. Allerdings bedient sich das Disc-Basic dabei einer Sprache, die dem Unbedarften zunächst einmal ein wenig "spanisch" vorkommen wird.
Um das Programm einwandfrei zu nutzen, sollten Sie sich erst einmal über den grundsätzlichen Aufbau einer Ihrer Disketten im klaren sein.
Was solche Zauberworte wie Track, Sektor oder Offset überhaupt bedeuten, soll Ihnen der folgende Bericht vermitteln.
Daß eine Diskette, bevor sie genutzt werden kann, erst formatiert werden muß, weiß jeder, denn es steht ja im Handbuch. Worüber sich das Handbuch jedoch ausschweigt: Wozu ist eine solche Formatierung überhaupt vonnöten und was geschieht dabei.

Während des Formatierens wird die Diskette zur Aufnahme von Daten vorbereitet. Das heißt, sie wird in einzelne Abschnitte und sogenannte Spuren unterteilt.
Eine auf der Diskette abgelegte physikalische Einheit zum Ablegen von Daten, besteht aus einem 512 Bytes großen Sektor. Insgesamt befinden sich dort 360 Sektoren, die sich über 40 Spuren verteilen. 360 Sektoren geteilt durch 40 Spuren ergibt eine Anzahl von 9 Sektoren pro Spur. Theoretisch ist die Diskette in der Lage, bis zur Spur 43 zu gehen. Diese unbenutzten Spuren lassen sich jedoch nicht von der internen Logik des Amsdos verwalten und können nur über Programmiertricks oder Hilfsprogramme wie Disc-Basic genutzt werden.

Jeder Sektor wird dabei noch einmal in vier logische Blöcke à 128 Bytes unterteilt. Ein solcher Block heißt Record und ist die kleinste Speichereinheit auf der Diskette. Auch wenn Sie weniger als 128 Bytes ablegen, ein Record wird immer voll genutzt.

Anders verhält es sich bei Dateien. Hier wird grundsätzlich ein 1 KB großer Bereich auf der Disk benötigt. Da sich ein Diskettenabschnitt von genau 1000 Bytes physikalisch nicht darstellen läßt, verwendet Amsdos dafür 2 Sektoren.

Ein 1 KB großer Block ist also in Wirklichkeit 2*512 Bytes = 1024 Bytes groß. 1024 Bytes sind die kleinste Speichereinheit, die Amsdos für Programme verwenden kann. Also auch, wenn Sie nur eine einzige Zeile mit dem Save- oder Openout-Kommando auf Diskette schreiben, ein 1 KB großer Block geht in jedem Fall verloren.

Diese 1 KB-Blöcke werden mit einer fortlaufenden Nummer belegt, an der man die genaue Position von Track und Sektor des Blocks bestimmen kann.

Eine genaue Formel, wie sich so etwas berechnet, finden Sie im Abschluß dieses Berichts.

Zum Arbeiten mit diesem Bericht nehmen Sie nun eine Leerdiskette und formatieren sie unter Disc-Basic durch Eingabe von:

IFORMAT,0

Nachdem die Systemmeldung 'FORMATING...PLEASE WAIT', abgeschlossen von READY, erscheint, haben Sie schon eine Diskette im Systemformat angelegt. Dieses Format soll die Grundlage für die weitere Arbeit sein. Alle Angaben dieses Berichts beziehen sich auf das Systemformat.

Ihre formatierte Diskette hat nun folgende Aufteilung:

SPUR 0/SEKTOR 0: BOOT SEKTOR (CP/M)
SPUR 0/SEKTOR 1: KONFIGURATIONSSEKTOR (CP/M)
SPUR 0/SEKTOR 2-6: UNBENUTZT
SPUR 0/SEKTOR 7-8: ANFANG DES CCP (CP/M)
SPUR 1/SEKTOR 0-8: ENDE CCP UND BDOS (CP/M)
SPUR 2/SEKTOR 0-4: DIRECTORY
SPUR 2/SEKTOR 5-8: ANFANG DATEN
SPUR 3-8/SEKTOR 0-8: ENDE DATEN

Die ersten beiden Spuren 0-1 können vom Anwender nicht genutzt werden, da sie dem CP/M vorbehalten sind. Man nennt sie Systemspuren.
Auf den ersten vier Sektoren der Spur 2 steht das Directory. Auch dieser Bereich sollte vom Anwender nicht verändert werden, da das Amsdos diesen Bereich benötigt, um sich auf der Diskette zurechtzufinden.
Nach dem Formatieren ist die Diskette komplett mit dem Filler-Byte &E5 gefüllt. Wir können das kontrollieren, indem wir einmal den Directorysektor der unbenutzten Diskette lesen. Geben Sie dazu unter Disc-Basic V3.0 ein:

IREAD,0,2,&41
IEXAMINE,0

Der Rechner gibt Ihnen in Mode 2 den in Abbildung 1 gezeigten Ausdruck auf den Bildschirm aus.

Abbildung 1

```
A000 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A00F E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A01E E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A02D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A03C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A04B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A05A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A069 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A078 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A087 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A096 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0A5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0B4 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0C3 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0D2 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0E1 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0F0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0FF E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A10E E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A11D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A12C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A13B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A14A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A159 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A168 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A177 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A186 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A195 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1A4 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1B3 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1C2 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1D1 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1E0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1EF E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1FE E5 E5 00 00 00 00 00 00 00 00 00 00 00 00 00  ee.............
```

Wie Sie sehen, ist dieser Track noch vollkommen unbenutzt. Saven Sie jetzt einmal eine Datei ab.
Dazu geben Sie ein:

```
10 REM DIES IST EIN TEST                          [ENTER]
20 REM UM ZU SEHEN WIE DIE DATEN                  [ENTER]
30 REM AUF DER DISKETTE ABGELEGT WERDEN           [ENTER]

SAVE"TEST"                                        [ENTER]
IREAD,0,2,&41                                     [ENTER]
IEXAMINE,0                                        [ENTER]
```

Fortsetzung auf S. 42

Disc Basic V3.0

Das Amsdos der Floppystation DDI-1 bietet über die bekannten Floppybefehle ziemlich wenig Möglichkeiten, eine vernünftige Diskettenprogrammierung vorzunehmen. Assemblerkenntnisse und eine gute Orientierung im Floppy-ROM, sind Voraussetzung dafür, möglichst effizient und sicher zu arbeiten.

Den Anwendern, denen M-Code und die Mysterien des siebten ROM's des CPC jedoch ein Buch mit sieben Siegeln sind, bleiben die, im Grunde genommen, fantastischen Möglichkeiten dieser, in der Anfangszeit als schneller Kassettenrekorder verschrieenen, Diskettenstation verschlossen.

Um Anfängern und Profis den Umgang mit den Routinen des Controllers zu ermöglichen bzw. zu erleichtern, veröffentlichen wir an dieser Stelle das Disc-Basic V3.0, welches ein ausgewogenes Maß an Befehlen bereithält. Insgesamt 20 neue Befehle für Hobby und professionelle Anwendung stehen nach Eingabe des ca. 1KB großen MC-Programmes zur Verfügung. Diese Befehle lassen sich in Gruppen unterteilen. Da sind zunächst einmal die Floppy-spezifischen Befehle wie:

READ, DRIVE, TRACK, SEKTOR
WRITE, DRIVE, TRACK, SEKTOR
SETBUF, ADRESSE
EXAMINE, DEVICE
FORMATEX, DRIVE, TRACK, OFFSET

Die nächste Gruppe von Befehlen stellt Instruktionen und Routinen bereit, die in Programmen recht nützlich sein können und bereits vorhandene Floppyoperationen verbessern.

FORMAT, FORMATOFFSET
FAST
SLOW
CLEAN
DELCAT
PCAT

Außerdem gibt es noch Hilfsroutinen, die zwar eigentlich nichts mit Floppyoperationen zu tun haben, jedoch die Programmentwicklung mit dem Disc-Basic V3.0 erleichtern.

BHELP
INIT
WAIT

Und als zusätzlichen Bonus für MC-kundige Programmierer einige Miniaturbefehle, die das Laden eines Assemblers oder Monitors bei Kurzroutinen ersparen sollen.

AKKU, BYTE
HL, BYTE, BYTE
BC, BYTE, BYTE
DE, BYTE, BYTE
SYS, ADRESSE
HDUMP, START, ENDE, DEVICE

Alle diese Befehle sind sogenannte RSXen (Resistent System Extensions) und müssen mit einem vorangestellten Strich, den Sie über die Tastenkombination **[SHIFT] [@]** erreichen, eingegeben werden. Will man beispielsweise den Akku mit dem Wert 7 laden, so ist die korrekte Syntax für den Befehl

|AKKU,7

Vergessen Sie den RSX-Strich, so wird die System-eigene Fehlermeldung SYNTAX ERROR ausgegeben, schreiben Sie den RSX-Befehl falsch, so erscheint UNKNOWN COMMAND und für das Weglassen eines unbedingt benötigten Parameters hat das Disc-Basic die eigene Fehlermeldung

!!! PARAMETERFEHLER !!!

Der Basiclader schreibt das Listing in den Speicherbereich &8000 und sichert ihn danach auf Diskette.

Mit folgendem kleinen Loader, kann das Basic danach direkt in den Speicher geladen, die Memory-Grenze gesetzt und die Initialisierungsroutine aufgerufen werden.

10 MEMORY &7FFF
20 LOAD "DB.BIN",&8000
30 CALL &8000
40 |BHELP

Im nachfolgenden werden die einzelnen Befehle genau erklärt, die Referenz zur Speicherbelegung unter Disc-Basic V3.0 sowie eine Routine, die dem Anwender hilft, auf einer mit |FORMATEX formatierten Diskette etwas wiederzufinden.

Befehlsbeschreibung:

1. READ
SYNTAX :|READ,DRIVE, TRACK,SEKTOR

FUNKTION:
Der Befehl dient zum Lesen eines Sektors auf Spur 0 bis 42 mit frei wählbarem Offset. Bei einer nicht formatierten Spur oder einem falschen Offset erfolgt die Disc-eigene Fehlerabfangroutine. Die gelesenen Daten werden in den Disc-Basic-Sektor Buffer geschrieben und haben eine Länge von 512 Bytes.
Ob die Spur vorhanden ist und mit welchem Offset sie gekennzeichnet ist, können Sie über das Hilfsprogramm SEKTOR MAP (Listing 2) ermitteln.

Beispiel:
Zum Lesen des Sektors &41 auf Spur 2 des Drives 0 geben Sie ein:
|READ,0,2,&42

2. WRITE

SYNTAX :|WRITE,DRIVE, TRACK,SEKTOR

FUNKTION:
Der Befehl dient zum Schreiben des Sektor-Buffers auf einen Sektor in Spur 0 bis 42 mit frei wählbarem Offset. Bei einer nicht formatierten Spur oder einem falschen Offset erfolgt die Disc-eigene Fehlerabfangroutine. Die zu schreibenden Daten haben eine Länge von 512 Bytes. Ob eine Spur vorhanden ist und mit welchem Offset sie gekennzeichnet ist, können Sie über das Hilfsprogramm SEKTOR MAP (Listing 3) ermitteln.

Beispiel:
Zum Schreiben des Buffers auf Sektor &41 der Spur 2 im Laufwerk 0, geben Sie ein:
|WRITE,0,2,&41

3. SETBUF
SYNTAX :|SETBUF, ADRESSE

FUNKTION:
Festlegen der Startadresse des 512 Bytes großen SEKTOR BUFFERS.
Beim Start des Disc-Basic steht dieser Buffer ab Adresse &A000.
In diesem Buffer werden die Lesedaten bei |READ und |WRITE abgelegt bzw. entnommen.

Beispiel:
Sie wollen eine Routine, die ab Adresse &5000 steht, auf Sektor &41, Spur 2, Laufwerk 0 schreiben, so geben Sie ein:
|SETBUF,&5000
|WRITE,0,2,&41

4. EXAMINE
SYNTAX :|EXAMINE, DEVICE

FUNKTION:
Ausgabe des momentanen SEKTOR BUFFER's auf die vorgewählte Device. Das Ausgabegerät kann entweder Monitor [0] oder Drucker [1] sein. Je nach angewähltem Mode, wird mit unterschiedlicher Spaltenbreite ausgegeben. Die Adresse der Bufferzeile wird mit ausgegeben. Neben den Hexwerten der

für 464-664-6128



```
1 '****************************************
2 '* DISCBASIC V3.0 BY CPC INTERNATIONAL *
3 '*--------------------------------------*
4 '*   COPYRIGHT 1986 BY THOMAS MORGEN    *
5 '****************************************
6 '
7 DATA &01,&0D,&80,&21,&09,&80,&C3,&D1,&BC,&F8,&A5
,&0D,&80,&4B,&80,&C3,&A6,&80,&C3,&BE,&80,&C3,&D6,&
80,&C3, 3395
8 DATA &E8,&80,&C3,&FB,&80,&C3,&0B,&81,&C3,&21,&81
,&C3,&2E,&81,&C3,&A7,&81,&C3,&6D,&82,&C3,&4D,&85,&
C3,&40, 3585
9 DATA &86,&C3,&57,&86,&C3,&5B,&86,&C3,&68,&86,&C3
,&B6,&86,&C3,&C8,&86,&C3,&DA,&86,&C3,&EC,&86,&C3,&
FF,&86, 4139
10 DATA &52,&45,&41,&C4,&57,&52,&49,&54,&C5,&53,&4
5,&54,&42,&55,&C6,&46,&41,&53,&D4,&53,&4C,&4F,&D7,
&44,&45, 2540
11 DATA &4C,&43,&41,&D4,&50,&43,&41,&D4,&46,&4F,&5
2,&4D,&41,&54,&45,&D8,&46,&4F,&52,&4D,&41,&D4,&42,
&48,&45, 2378
12 DATA &4C,&D0,&45,&58,&41,&4D,&49,&4E,&C5,&43,&4
C,&45,&41,&CE,&57,&41,&49,&D4,&49,&4E,&49,&D4,&48,
&44,&55, 2512
13 DATA &4D,&D0,&48,&CC,&44,&C5,&42,&C3,&41,&4B,&4
B,&D5,&53,&59,&D3,&00,&FE,&03,&C2,&15,&86,&2A,&E6,
&80,&DD, 3125
14 DATA &5E,&04,&DD,&56,&02,&DD,&4E,&00,&DF,&BA,&8
0,&66,&C6,&07,&C9,&FE,&03,&C2,&15,&86,&2A,&E6,&80,
&DD,&5E, 3072
15 DATA &04,&DD,&56,&02.&DD.&4E,&00,&DF,&D2,&80,&4
E,&C6,&07,&C9,&FE,&01,&C2,&15,&86,&DD,&56,&01,&DD,
```

```
&5E,&00, 2884
16 DATA &ED,&53,&E6,&80,&C9,&00,&A0,&21,&F2,&80,&D
F,&EE,&80,&0D,&C6,&07,&C9,&23,&00,&C8,&00,&01,&01,
&0A,&00, 2697
17 DATA &03,&21,&02,&81,&DF,&EE,&80,&C9,&32,&00,&F
A,&00,&AF,&0F,&0C,&01,&03,&3E,&0C,&CD,&5A,&BB,&3E,
&E5,&32, 2360
18 DATA &01,&A7,&11,&00,&90,&CD,&9B,&BC,&3E,&00,&3
2,&01,&A7,&C9,&CD,&EB,&85,&11,&00,&90,&CD,&9B,&BC,
&CD,&00, 2845
19 DATA &86,&C9,&FE,&03,&C2,&15,&86,&DD,&5E,&04,&D
D,&56,&02,&DD,&7E,&00,&32,&80,&81,&21,&81,&81,&06,
&09,&7A, 2651
20 DATA &77,&23,&23,&23,&23,&10,&F8,&3A,&80,&81,&3
2,&83,&81,&3C,&32,&97,&81,&3C,&32,&87,&81,&3C,&32,
&9B,&81, 2306
21 DATA &3C,&32,&8B,&81,&3C,&32,&9F,&81,&3C,&32,&8
F,&81,&3C,&32,&A3,&81,&3C,&32,&93,&81,&3A,&80,&81,
&4F,&21, 2373
22 DATA &81,&81,&DF,&7D,&81,&C9,&52,&C6,&07,&41,&0
0,&00,&41,&02,&00,&00,&43,&02,&00,&00,&45,&02,&00,
&00,&47, 1566
23 DATA &02,&00,&00,&49,&02,&00,&00,&42,&02,&00,&0
0,&44,&02,&00,&00,&46,&02,&00,&00,&48,&02,&00,&00,
&FE,&01, 616
24 DATA &C2,&15,&86,&DD,&22,&1E,&82,&21,&00,&82,&C
D,&18,&86,&3E,&00,&32,&1F,&82,&3A,&1F,&82,&57,&3A,
&20,&82, 2089
25 DATA &5F,&3A,&21,&82,&4F,&3A,&1E,&82,&FE,&01,&2
8,&05,&21,&25,&82,&18,&03,&21,&49,&82,&DF,&22,&82,
&3A,&1F, 1852
26 DATA &82,&FE,&27,&C8,&3C,&32,&1F,&82,&3A,&1E,&8
2,&FE,&01,&28,&05,&21,&25,&82,&18,&03,&21,&49,&82,
&06,&09, 1890
27 DATA &3A,&1F,&82,&77,&23,&23,&23,&23,&10,&F6,&1
8,&BB,&07,&0A,&0D,&46,&4F,&52,&4D,&41,&54,&54,&49,
&4E,&47, 1744
28 DATA &2E,&2E,&2E,&50,&4C,&45,&41,&53,&45,&20,&5
7,&41,&49,&54,&0A,&0D,&00,&00,&27,&00,&41,&52,&C6,
&07,&00, 1335
29 DATA &00,&41,&02,&00,&00,&43,&02,&00,&00,&45,&0
2,&00,&Q0,&47,&02,&00,&00,&49,&02,&00,&00,&42,&02,
&00,&00, 423
30 DATA &44,&02,&00,&00,&46,&02,&00,&00,&48,&02,&0
0,&00,&C1,&02,&00,&00,&C3,&02,&00,&00,&C5,&02,&00,
&00,&C7, 1006
31 DATA &02,&00,&00,&C9,&02,&00,&00,&C2,&02,&00,&0
0,&C4,&02,&00,&00,&C6,&02,&00,&00,&C8,&02,&21,&7C,
&82,&7E, 1414
32 DATA &FE,&00,&C8,&CD,&5A,&BB,&23,&C3,&70,&82,&C
9,&0C,&18,&44,&49,&53,&43,&42,&41,&53,&49,&43,&20,
&56,&31, 2457
33 DATA &2E,&30,&20,&42,&59,&20,&54,&4D,&20,&2E,&2
E,&2E,&20,&48,&45,&4C,&50,&2D,&50,&41,&47,&45,&18,
&0A,&0A, 1347
34 DATA &0D,&7C,&52,&45,&41,&44,&2E,&2E,&2E,&2E,&2
C,&5B,&44,&52,&49,&56,&45,&5D,&2C,&5B,&54,&52,&41,
&43,&4B, 1719
35 DATA &5D,&2C,&5B,&53,&45,&4B,&54,&4F,&52,&5D,&0
A,&0D,&7C,&57,&52,&49,&54,&45,&2E,&2E,&2E,&2C,&5B,
&44,&52, 1758
36 DATA &49,&56,&45,&5D,&2C,&5B,&54,&52,&41,&43,&4
B,&5D,&2C,&5B,&53,&45,&4B,&54,&4F,&52,&5D,&0A,&0D,
&7C,&53, 1852
37 DATA &45,&54,&42,&55,&46,&2E,&2E,&2C,&5B,&41,&4
4,&52,&45,&53,&53,&45,&5D,&0A,&0D,&7C,&48,&44,&55,
&4D,&50, 1742
38 DATA &2E,&2E,&2E,&2C,&5B,&53,&54,&41,&52,&54,&5
D,&2C,&5B,&45,&4E,&44,&45,&5D,&2C,&5B,&58,&5D,&0A,
&0D,&7C, 1739
39 DATA &46,&41,&53,&54,&2E,&2E,&2E,&2E,&2E,&3C,&5
4,&55,&52,&42,&4F,&20,&44,&4F,&53,&20,&4F,&4E,&3E,
&0A,&0D, 1524
40 DATA &7C,&53,&4C,&4F,&57,&2E,&2E,&2E,&2E,&2E,&3
C,&54,&55,&52,&42,&4F,&20,&44,&4F,&53,&20,&4F,&46,
&46,&3E, 1710
41 DATA &0A,&0D,&7C,&57,&41,&49,&54,&2E,&2E,&2E,&2
E,&2E,&3C,&57,&41,&52,&54,&45,&20,&41,&55,&46,&20,
&54,&41, 1566
42 DATA &53,&54,&45,&4E,&44,&52,&55,&43,&4B,&3E,&0
A,&0D,&7C,&49,&4E,&49,&54,&2E,&2E,&2E,&2E,&2E,&3C,
&4B,&45, 1642
43 DATA &59,&42,&4F,&41,&52,&44,&2C,&42,&49,&4C,&4
4,&53,&43,&48,&49,&52,&4D,&20,&49,&4E,&49,&54,&3E,
&0A,&0D, 1655
44 DATA &7C,&44,&45,&4C,&43,&41,&54,&2E,&2E,&2E,&3
C,&41,&55,&53,&47,&41,&42,&45,&20,&47,&45,&4C,&4F,
&45,&53, 1734
45 DATA &43,&48,&54,&45,&52,&20,&46,&49,&4C,&45,&5
3,&3E,&0A,&0D,&7C,&50,&43,&41,&54,&2E,&2E,&2E,&2E,
&2E,&3C, 1572
46 DATA &44,&49,&52,&45,&43,&54,&4F,&52,&59,&20,&4
1,&55,&46,&20,&44,&52,&55,&43,&4B,&45,&52,&3E,&0A,
&0D,&7C, 1714
47 DATA &46,&4F,&52,&4D,&41,&54,&45,&58,&2C,&5B,&4
4,&52,&49,&56,&45,&5D,&2C,&5B,&54,&52,&41,&43,&4B,
&5D,&2C, 1865
48 DATA &5B,&4F,&46,&46,&53,&45,&54,&5D,&0A,&0D,&7
C,&46,&4F,&52,&4D,&41,&54,&2E,&2E,&2C,&5B,&58,&5D,
&20,&20, 1715
49 DATA &3C,&30,&2D,&56,&45,&4E,&44,&4F,&52,&2F,&3
1,&2D,&44,&41,&54,&41,&20,&4F,&4E,&4C,&59,&3E,&0A,
&0D,&7C, 1601
50 DATA &43,&4C,&45,&41,&4E,&2E,&2E,&2E,&2E,&3C,&4
C,&4F,&45,&53,&43,&48,&54,&20,&41,&4C,&4C,&45,&20,
&42,&41, 1610
51 DATA &4B,&20,&44,&41,&54,&45,&49,&45,&4E,&3E,&0
A,&0D,&7C,&42,&48,&45,&4C,&50,&2E,&2E,&2E,&2E,&3C,
&4C,&49, 1578
52 DATA &53,&54,&45,&20,&44,&45,&52,&20,&42,&45,&4
6,&45,&48,&4C,&45,&3E,&0A,&0D,&7C,&45,&58,&41,&4D,
&49,&4E, 1669
53 DATA &45,&20,&2C,&5B,&58,&5D,&20,&20,&3C,&30,&2
D,&4D,&4F,&4E,&49,&54,&4F,&52,&2F,&31,&2D,&44,&52,
&55,&43, 1629
54 DATA &4B,&45,&52,&3E,&0A,&0D,&7C,&48,&4C,&2E,&2
E,&2E,&2E,&2E,&2E,&2C,&5B,&58,&5D,&2C,&5B,&59,&5D,
&20,&20, 1556
55 DATA &3C,&48,&4C,&20,&52,&45,&47,&49,&53,&54,&4
5,&52,&20,&4C,&41,&44,&45,&4E,&3E,&0A,&0D,&7C,&44,
&45,&2E, 1633
56 DATA &2E,&2E,&2E,&2E,&2E,&2C,&5B,&58,&5D,&2C,&5
B,&59,&5D,&20,&20,&3C,&44,&45,&20,&52,&45,&47,&49,
&53,&54, 1618
57 DATA &45,&52,&20,&4C,&41,&44,&45,&4E,&3E,&0A,&0
D,&7C,&42,&43,&2E,&2E,&2E,&2E,&2E,&2E,&2C,&5B,&58,
&5D,&2C, 1517
58 DATA &5B,&59,&5D,&20,&20,&3C,&42,&43,&20,&52,&4
5,&47,&49,&53,&54,&45,&52,&20,&4C,&41,&44,&45,&4E,
&3E,&0A, 1635
59 DATA &0D,&7C,&41,&4B,&4B,&55,&2E,&2E,&2E,&2E,&2
C,&5B,&58,&5D,&20,&20,&20,&20,&20,&20,&3C,&41,&4B,
&4B,&55, 1489
60 DATA &20,&4C,&41,&44,&45,&4E,&3E,&0A,&0D,&7C,&5
3,&59,&53,&2E,&2E,&2E,&2E,&2E,&2C,&5B,&41,&44,&52,
&45,&53, 1584
61 DATA &53,&45,&5D,&20,&0A,&0D,&00,&FE,&01,&C2,&1
5,&86,&3E,&0C,&CD,&5A,&BB,&DD,&7E,&00,&FE,&01,&CC,
&EB,&85, 2634
62 DATA &2A,&E6,&80,&11,&00,&02,&19,&22,&CC,&85,&2
A,&E6,&80,&CD,&91,&86,&7C,&CD,&CE,&85,&7D,&CD,&CE,
&85,&3E, 3098
63 DATA &20,&CD,&5A,&BB,&3A,&B4,&86,&47,&7E,&CD,&C
E,&85,&3E,&20,&CD,&5A,&BB,&23,&10,&F4,&3E,&20,&CD,
&5A,&BB, 3074
64 DATA &ED,&5B,&B4,&86,&B7,&ED,&52,&3A,&B4,&86,&4
7,&7E,&23,&CB,&BF,&FE,&20,&30,&02,&3E,&2E,&CD,&5A,
&BB,&10, 3084
65 DATA &F1,&3E,&0D,&CD,&5A,&BB,&3E,&0A,&CD,&5A,&B
B,&E5,&ED,&5B,&CC,&85,&B7,&ED,&52,&E1,&38,&AF,&28,
&AD,&3A, 3475
66 DATA &CB,&85,&FE,&01,&CC,&00,&86,&C9,&00,&00,&0
0,&5F,&0F,&0F,&0F,&0F,&E6,&0F,&CD,&DF,&85,&7B,&E6,
&0F,&CD, 2664
67 DATA &DF,&85,&C9,&FE,&0A,&38,&02,&C6,&07,&C6,&3
0,&CD,&5A,&BB,&C9,&3E,&01,&32,&CB,&85,&3E,&C3,&32,
&5A,&BB, 3046
68 DATA &3E,&2B,&32,&5B,&BB,&3E,&BD,&32,&5C,&BB,&C
9,&3E,&00,&32,&CB,&85,&3E,&CF,&32,&5A,&BB,&3E,&00,
&32,&5B, 2461
69 DATA &BB,&3E,&94,&32,&5C,&BB,&C9,&21,&23,&86,&7
E,&FE,&00,&C8,&CD,&5A,&BB,&23,&C3,&18,&86,&0A,&0D,
&21,&21, 2668
70 DATA &21,&20,&50,&41,&52,&41,&4D,&45,&54,&45,&5
2,&46,&45,&48,&4C,&45,&52,&20,&21,&21,&21,&07,&0A,
&0D,&00, 1337
71 DATA &3E,&01,&DD,&21,&4D,&86,&DF,&4A,&86,&C9,&2
D,&C0,&07,&4F,&86,&05,&52,&86,&2A,&2E,&42,&41,&4B,
&CD,&06, 2343
72 DATA &BB,&C9,&CD,&00,&BB,&CD,&4E,&BB,&CD,&BA,&B
B,&CD,&BF,&BB,&C9,&FE,&03,&C2,&15,&86,&3E,&0C,&CD,
&5A,&BB, 3774
73 DATA &DD,&7E,&00,&FE,&01,&CC,&EB,&85,&DD,&56,&0
3,&DD,&5E,&02,&ED,&53,&CC,&85,&DD,&66,&05,&DD,&6E,
&04,&CD, 3326
74 DATA &91,&86,&CD,&6F,&85,&C9,&CD,&11,&BC,&FE,&0
2,&CA,&AE,&86,&FE,&01,&CA,&A6,&86,&3E,&03,&32,&B4,
&86,&C3, 3486
```

```
75 DATA &B3,&86,&3E,&08,&32,&B4,&86,&C3,&B3,&86,&3
E,&0F,&32,&B4,&86,&C9,&00,&00,&FE,&02,&C2,&15,&86,
&DD,&7E, 2849
76 DATA &02,&32,&F8,&86,&DD,&7E,&00,&32,&F9,&86,&C
9,&FE,&02,&C2,&15,&86,&DD,&7E,&02,&32,&FA,&86,&DD,
&7E,&00, 3150
77 DATA &32,&FB,&86,&C9,&FE,&02,&C2,&15,&86,&DD,&7
E,&02,&32,&FC,&86,&DD,&7E,&00,&32,&FD,&86,&C9,&FE,
&01,&C2, 3460
78 DATA &15,&86,&DD,&7E,&00,&32,&FE,&86,&C9,&00,&0
0,&00,&00,&00,&00,&00,&FE,&01,&C2,&15,&86,&DD,&56,
&01,&DD, 2274
79 DATA &5E,&00,&ED,&53,&30,&87,&3A,&F8,&86,&67,&3
A,&F9,&86,&6F,&3A,&FA,&86,&57,&3A,&FB,&86,&5F,&3A,
&FC,&86, 3203
80 DATA &47,&3A,&FD,&86,&4F,&3A,&FE,&86,&DD,&2A,&3
0,&87,&DD,&E9,&C9,&01,&AC,&00, 2315
81 dat=0 : sz=0 : dz = 1
82 FOR adr =-32768  TO-30926
83   READ byte : dat=dat+1
84   sz=sz+byte
85   POKE adr,byte
86   IF dat < 25 AND adr < -30926 THEN 90
87   READ chksum
88   IF chksum<>sz THEN PRINT "Fehler in Zeile :";
dz
89   dz=dz + 1 : sz=0 : dat=0
90 NEXT adr
91 SAVE"DB",B,&8000,&732
92 END
```

Adressen, werden am rechten Bildschirmrand auch noch die entsprechenden ASCII-Zeichen ausgegeben.

Beispiel:
Der Directorysektor einer, im Data-Format beschriebenen Diskette, soll auf Drucker mit 16 Zeichen Breite ausgedruckt werden. Sie geben ein:

MODE 2
IREAD,0,0,&C1
IEXAMINE,1

5. FORMATEX
SYNTAX :IFORMATEX, DRIVE,TRACK,OFFSET

FUNKTION:
Formatieren eines einzelnen Tracks mit beliebigem Offset. Mit diesem Befehl sollte man vorsichtig umgehen – überformatierte Daten sind unwiederruflich verloren. Allerdings ist diese Routine auch sehr nützlich beim Retten von Disketten, auf denen eine Spur durch Magnetismus oder ähnliches verlorengegangen ist. Die Tracks (Spuren) können im Bereich 0 - 42 liegen. Auch zum Schützen von Disketten ist dieser Befehl sehr nützlich. Eine nicht formatierte Spur, oder eine mit falschem Offset läßt sich nicht mehr so ohne weiteres kopieren. Der Offset wird vom Anwender bestimmt. Standard-Offsets sind &41 [Systemformat-CP/M], &C1 [Data Only] und &01 [IBM Format]. Die Lage der Sektoren auf der Spur wird vom Programm bestimmt. Hier sind keine Manipulationen möglich. Die Ablage erfolgt nach Standard und sieht am Beispiel IBM Format folgendermaßen aus:

&01 &06 &03 &07 &04 &08 &05 &09

Beispiel:
Zum Formatieren einer Diskette im IBM Format bis zur Spur 42 geben Sie ein:

10 FOR TRACK = 0 TO 42
20 PRINT TRACK
30 IFORMATEX,0,TRACK, &01
40 NEXT TRACK
RUN [ENTER]

```
;*******************************
;** DISCBASIC V3.0 BY CPC INT **
;**                          **
;** 1986  THOMAS MORGEN      **
;**                          **
;*******************************
ORG &8000                      ;STARTADRESSE
LD BC,RSX                      ;SPRUNGTABELLE
LD HL,KERNAL                   ;RSX BUFFER
JP &BCD1                       ;LOG EXT
KERNAL:DEFS 4                  ;4 BYTE RSX BUFFER
;-------------------------------
RSX:DEFW TABLE                 ;SPRUNGTABELLE
JP LESE
JP SCHREIBE
JP LESBUF
JP FAST
JP SLOW
JP DELCAT
JP PCAT
JP FORMATEX
JP FORMAT
JP HELP
JP AUSGA
JP CLEAN
JP WARTE
JP REDEF
JP MINIMON
JP HREG
JP DREG
JP BREG
JP AREG
JP SYSCA
;--------------------------------
TABLE:DEFM "REA"                  ;BEFEHLSWORTE
DEFB "D"+&80
DEFM "WRIT"
DEFB "E"+&80
DEFM "SETBU"
DEFB "F"+&80
DEFM "FAS"
DEFB "T"+&80
DEFM "SLO"
DEFB "W"+&80
DEFM "DELCA"
DEFB "T"+&80
DEFM "PCA"
DEFB "T"+&80
DEFM "FORMATE"
DEFB "X"+&80
DEFM "FORMA"
DEFB "T"+&80
DEFM "BHEL"
DEFB "P"+&80
DEFM "EXAMIN"
DEFB "E"+&80
DEFM "CLEA"
DEFB "N"+&80
DEFM "WAI"
DEFB "T"+&80
DEFM "INI"
DEFB "T"+&80
DEFM "HDUM"
DEFB "P"+&80
DEFB "H"
DEFB "L"+&80
DEFB "D"
DEFB "E"+&80
DEFB "B"
DEFB "C"+&80
DEFM "AKK"
DEFB "U"+&80
DEFM "SY"
DEFB "S"+&80
NOP
```

```
;------------------------------- *** SUB SEKTOR LESEN ***
LESE:CP 3                        ;DREI PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD HL,(BUFFER)                   ;BUFFER NACH HL
LD E,(IX+4)                      ;ERSTER PARAMETER (DRIVE)
LD D,(IX+2)                      ;ZWEITER PARAMETER (TRACK)
LD C,(IX)                        ;DRITTER PARAMETER (SECTOR)
RST &18,LESEADR                  ;READ SECTOR
LESEADR: DEFB &66,&C6,&07        ;ROM 7 - ADRESSE &C666
RET                              ;ZURUECK
;------------------------------- *** SUB SEKTOR SCHREIBEN ***
SCHREIBE: CP 3                   ;DREI PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD HL,(BUFFER)                   ;BUFFER NACH HL
LD E,(IX+4)                      ;ERSTER PARAMETER (DRIVE)
LD D,(IX+2)                      ;ZWEITER PARAMETER (TRACK)
LD C,(IX)                        ;DRITTER PARAMETER (SECTOR)
RST &18,SCHREIBADR               ;WRITE SEKTOR
SCHREIBADR: DEFB &4E,&C6,&07     ;ROM 7 - ADRESSE &C64E
RET                              ;ZURUECK
;------------------------------- *** SUB BUFFER SETZEN ***
LESBUF: CP 1                     ;EIN PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD D,(IX+1)                      ;BYTE 1 ADRESSE NACH D
LD E,(IX)                        ;BYTE 2 ADRESSE NACH E
LD (BUFFER),DE                   ;BUFFER = DE
RET                              ;ZURUECK
BUFFER:DEFB &00,&A0              ;BUFFER POINTER DEFAULT &A000
;------------------------------- *** SUB FAST ***
FAST:LD HL,NEWTAB                ;TIMINGTABELLE (NEU)
RST &18,NEUADR                   ;TIMING SETZEN
NEUADR: DEFB &0D,&C6,&07         ;ROM 7 - ADRESSE &C60D
RET                              ;ZURUECK
NEWTAB: DEFB &23,&00,&C8,&00     ;NEUE TIMINGTABELLE
        DEFB &01,&01,&0A,&00     ;PART 2
        DEFB &03                 ;PART 3
;------------------------------- *** SUB SLOW ***
SLOW:LD HL,OLDTAB                ;TIMINGTABELLE (ALT)
RST &18,NEUADR                   ;TIMING SETZEN
RET                              ;ZURUECK
OLDTAB: DEFB &32,&00,&FA,&00     ;ALTE TIMINGTABELLE
        DEFB &AF,&0F,&0C,&01     ;PART 2
        DEFB &03                 ;PART 3
;------------------------------- *** SUB DELCAT ***
DELCAT:LD A,&0C                  ;CHAR FOR CLS
CALL &BB5A                       ;CLS AUSFUEHREN
LD A,&E5                         ;AKKU = 229
LD (&A701),A                     ;USER BEREICH AUF 229
LD DE,&9000                      ;DIRBUFFER BEI &9000
CALL &BC9B                       ;CATALOG
LD A,&00                         ;AKKU = 0
LD (&A701),A                     ;USER BEREICH AUF 0
RET                              ;ZURUECK
;------------------------------- *** SUB PCAT ***
PCAT:CALL SWAP                   ;VEKTOR EINRICHTEN
ld de,&9000                      ;DIRBUFFER BEI &9000
CALL &BC9B                       ;CATALOG
CALL SWAPOFF                     ;VEKTOR RESTAURIEREN
RET                              ;ZURUECK
;------------------------------- *** SUB FORMATEX ***
FORMATEX:CP 3                    ;3 PARAMETER ?
JP NZ,FEHLER                     ;ZURUECK IF NOT
LD E,(IX+4)                      ;DRITTER PARAMETER [DRIVE]
LD D,(IX+2)                      ;ZWEITER PARAMETER [TRACK]
LD A,(IX)                        ;OFFSET NACH A
LD (OFLAG),A                     ;ERSTER  PARAMETER [OFFSET]
LD HL,ETABE                      ;TABELLE NACH HL
LD B,09                          ;NINE TIMES
SELOOP:LD A,D                    ;TRACK NACH A
LD (HL),A                        ;TRACK NACH HL
INC HL                           ;FOUR TIMES TRACK SELECT IN TABELLE
INC HL                           ;HL = HL + 1
INC HL                           ;HL = HL + 1
INC HL                           ;HL = HL + 1
DJNZ SELOOP                      ;SCHLEIFE ZURUECK
LD A,(OFLAG)                     ;OFFSET NACH A
LD (SEC1),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC6),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC2),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC7),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC3),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC8),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC4),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
LD (SEC9),A                      ;OFFSET IN TABELLE
INC A                            ;A = A + 1
```

6. FORMAT
SYNTAX :IFORMAT,FORMATOFFSET
FUNKTION:
Der Befehl Format stellt eine von Basic aus aufrufbare Routine zur Verfügung, die eine Diskette, je nach gewähltem Offset, komplett formatiert und somit das Booten von CP/M abdingbar macht. Das Formatoffset kann im Data Only [1] oder Vendor Format [0] erfolgen.
!! ACHTUNG !!
Diese Routine hat im Gegensatz zu FORMAT.COM ihrer CP/M-Masterdiskette keine Sicherheitsabfrage und legt sofort nach Eingabe los.
Beispiel:
Eine Diskette soll im Vendor Format formatiert werden. Sie geben ein:
IFORMAT,0

7. FAST
SYNTAX :IFAST
FUNKTION:
Dieser Befehl spezifiziert die Timingtabelle der Floppy neu und ermöglicht einen bis zu 20 % schnelleren Zugriff auf alle Floppyoperationen.

8. SLOW
SYNTAX :ISLOW
FUNKTION:
Wiederherstellen der alten Timingtabelle nach IFAST.

9. CLEAN
SYNTAX :ICLEAN
FUNKTION:
Der Befehl dient dazu, auf einer überfüllten Diskette Platz zu schaffen, indem alle BAK-Dateien gelöscht werden.

10. DELCAT
SYNTAX :IDELCAT
FUNKTION:
Der Befehl setzt die Usernummer des Drives auf &E5 und führt einen Catalog aus. Auf der USER befinden sich alle gelöschten Files. Mit **POKE &A701,&E5:LOAD "DATEINAME":POKE &A701,0:SAVE "DATEINAME" [ENTER]** kann ein irrtümlich gelöschtes File wieder in das normale Directory gebracht werden.

11. PCAT
SYNTAX :IPCAT
FUNKTION:
Der Befehl verbiegt die Bildschirmausgaberoutine auf den Drucker, führt ein CAT aus und stellt den Screen Vektor danach wieder zurück. Dadurch erhält man einen Ausdruck des über IUSER angewählten Directorys auf

```
LD (SEC5),A                    ;OFFSET IN TABELLE
LD A,(OFLAG)                   ;OFFSET ZURUECK
LD C,A                         ;OFFSET AUF SEKTOR 1
LD HL,ETABE                    ;FORMAT TABELLE
RST &18,EFORM                  ;FORMATIERE TRACK
RET                            ;ZURUECK
;------------------------------ *** SUB TABELLE UND FLAGS FUER FORMATEX ***
EFORM:DEFB &52,&C6,&07         ;ROM 7, ADRESSE &C652
OFLAG:DEFB &41                 ;OFFSET DEFAULT &41
ETABE:DEFB &00,&00             ;TABELLE FUER FORMATEX
SEC1:DEFB &41,&02,&00,&00      ;-SEKTOR 1
SEC2:DEFB &43,&02,&00,&00      ;-SEKTOR 2
SEC3:DEFB &45,&02,&00,&00      ;-SEKTOR 3
SEC4:DEFB &47,&02,&00,&00      ;-SEKTOR 4
SEC5:DEFB &49,&02,&00,&00      ;-SEKTOR 5
SEC6:DEFB &42,&02,&00,&00      ;-SEKTOR 6
SEC7:DEFB &44,&02,&00,&00      ;-SEKTOR 7
SEC8:DEFB &46,&02,&00,&00      ;-SEKTOR 8
SEC9:DEFB &48,&02,&00,&00      ;-SEKTOR 9
;------------------------------ *** SUB FORMAT ***
FORMAT:cp 1                    ;EIN PARAMETER ?
JP NZ,FEHLER                   ;PARAMETERFEHLER IF NOT
LD (VFLAG),IX                  ;PARAMETER FLAG SETZEN
LD HL,WAIT                     ;BITTE WARTEN
CALL TXTOUT                    ;AUSGEBEN
LD A,0                         ;STARTTRACK = 0
LD (VFLAG1),A                  ;FLAG MOMENTANER TRACK
REPEAT:LD A,(VFLAG1)           ;MOMENTANER TRACK NACH A
LD D,A                         ;UND IN REGISTER D
LD A,(VFLAG2)                  ;FLAG FUER DRIVE SELECT
LD E,A                         ;IN REGISTER E
LD A,(VFLAG3)                  ;FLAG FUER SEKTOROFFSET
LD C,A                         ;IN REGISTER C
LD A,(VFLAG)                   ;PARAMETER NACH A
CP 1                           ;WAR 1 GEWAEHLT ?
JR Z,SKIP                      ;JA, SPRINGE NACH SKIP
LD HL,TABEL1                   ;NEIN, LADE TABELLE 1
JR AUFRUF                      ;NEIN, SPRINGE ROMCALL
SKIP:LD HL,TABEL2              ;JA, LADE TABELLE 2
AUFRUF:RST &18,FORMCA          ;FORMATIERE TRACK
LD A,(VFLAG1)                  ;TRACK NCH A
CP &27                         ;GROESSER &27 ?
RET Z                          ;ZURUECK IF TRUE
INC A                          ;ERHOEHE A IF NOT
LD (VFLAG1),A                  ;TRACK = TRACK + 1
LD A,(VFLAG)                   ;PARAMETER NACH A
CP 1                           ;WAR 1 GEWAEHLT ?
JR Z,SKIP2                     ;JA, SPRINGE NACH SKIP2
LD HL,TABEL1                   ;NEIN, LADE TABELLE 1
JR AUFRUF2                     ;NEIN, SPRINGE AUFRUF2
SKIP2:LD HL,TABEL2             ;JA, LADE TABELLE 2
AUFRUF2:LD B,9                 ;ANZAHL SEKTOREN = 9
VLOOP:LD A,(VFLAG1)            ;TRACK NACH A
LD (HL),A                      ;TRACK NACH HL
INC HL                         ;FOUR TIMES
INC HL                         ;HL = HL + 1
INC HL                         ;HL = HL + 1
INC HL                         ;HL = HL + 1
DJNZ VLOOP                     ;9 TIMES SCHLEIFE
JR REPEAT                      ;NAECHSTER TRACK
;------------------------------ *** SUB WARTETEXT ***
WAIT:DEFB &07,&0A,&0D          ;KLINGEL LF/CR
DEFM "FORMATTING...PLEASE WAIT";WARTETEXT
DEFB &0A,&0D,&00               ;LF/CR ENDE TABELLE
;------------------------------ *** SUB FORMAT FLAGS ***
VFLAG: DEFB 00                 ;PARAMETER FLAG
VFLAG1: DEFB &27               ;TRACK FLAG
VFLAG2: DEFB 00                ;DRIVE SELECT FLAG
VFLAG3: DEFB &41               ;SECTOR OFFSET FLAG
FORMCA: DEFB &52,&C6,&07       ;ROM 7 ADRESSE &C652
;------------------------------ *** SUB FORMAT TABELLE 1 [VENDOR] ***
TABEL1: DEFB &00,&00,&41,&02   ;- SECTOR 1
        DEFB &00,&00,&43,&02   ;- SECTOR 2
        DEFB &00,&00,&45,&02   ;- SECTOR 3
        DEFB &00,&00,&47,&02   ;- SECTOR 4
        DEFB &00,&00,&49,&02   ;- SECTOR 5
        DEFB &00,&00,&42,&02   ;- SECTOR 6
        DEFB &00,&00,&44,&02   ;- SECTOR 7
        DEFB &00,&00,&46,&02   ;- SECTOR 8
        DEFB &00,&00,&48,&02   ;- SECTOR 9
;------------------------------ *** SUB FORMAT TABELLE 2 [DATA ONLY] ***
TABEL2: DEFB &00,&00,&C1,&02   ;-SECTOR 1
        DEFB &00,&00,&C3,&02   ;-SECTOR 2
        DEFB &00,&00,&C5,&02   ;-SECTOR 3
        DEFB &00,&00,&C7,&02   ;-SECTOR 4
        DEFB &00,&00,&C9,&02   ;-SECTOR 5
        DEFB &00,&00,&C2,&02   ;-SECTOR 6
        DEFB &00,&00,&C4,&02   ;-SECTOR 7
        DEFB &00,&00,&C6,&02   ;-SECTOR 8
        DEFB &00,&00,&C8,&02   ;-SECTOR 9
;------------------------------ *** SUB HELP ***
HELP:LD HL,MESS                ;TEXT IN HL
```

```
LOOP:LD A,(HL)                    ;CHAR TO A
CP 0                              ;END OF TEXT ?
RET Z                             ;ZURUECK IF TRUE
CALL &BB5A                        ;CHAR TO BILDSCHIRM
INC HL                            ;HL = HL - 1
JP LOOP                           ;AUSGABE FORTSETZEN
RET                               ;ZURUECK
;-------------------------------- *** TEXT-TABELLE ***
MESS: DEFB &0C                                        ;CLS
DEFB &18                                              ;REVERS
DEFM "DISCBASIC V1.0 BY TM ... HELP-PAGE"             ;TITEL
DEFB &18,&0A,&0A,&0D                                  ;REVERS LF/LF/CR
DEFM "|READ....,[DRIVE],[TRACK],[SEKTOR]"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|WRITE...,[DRIVE],[TRACK],[SEKTOR]"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|SETBUF..,[ADRESSE]"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|HDUMP...,[START],[ENDE],[X]"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|FAST.....<TURBO DOS ON>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|SLOW.....<TURBO DOS OFF>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|WAIT.....<WARTE AUF TASTENDRUCK>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|INIT.....<KEYBOARD,BILDSCHIRM INIT>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|DELCAT...<AUSGABE GELOESCHTER FILES>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|PCAT.....<DIRECTORY AUF DRUCKER>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|FORMATEX,[DRIVE],[TRACK],[OFFSET]"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|FORMAT..,[X]  <0-VENDOR/1-DATA ONLY>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|CLEAN....<LOESCHT ALLE BAK DATEIEN>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|BHELP....<LISTE DER BEFEHLE>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|EXAMINE ,[X]  <0-MONITOR/1-DRUCKER>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|HL.......,[X],[Y]  <HL REGISTER LADEN>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|DE.......,[X],[Y]  <DE REGISTER LADEN>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|BC.......,[X],[Y]  <BC REGISTER LADEN>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|AKKU....,[X]       <AKKU LADEN>"
DEFB &0A,&0D                                          ;LF/CR
DEFM "|SYS.....,[ADRESSE] "
DEFB &0A,&0D,&00                                      ;LF/CR UND ENDE TABELLE
;-------------------------------- *** SUB EXAMINE ***
AUSGA:CP 1                        ;EIN PARAMETER ?
JP NZ,FEHLER                      ;PARAMETERFEHLER IF NOT
LD A,&0C                          ;CHAR FUER CLS
CALL &BB5A                        ;AUSGEBEN
LD A,(IX)                         ;ERSTER PARAMETER [STREAMNUMMER]
CP 1                              ;<> 1
CALL Z,SWAP                       ;VEKTOR EINRICHTEN
ld hl,(buffer)                    ;BUFFERSTART
LD DE,512                         ;LAENGE FESTLEGEN
ADD HL,DE                         ;ZU BUFFERSTART ADDIEREN
LD (SCHLUSS),HL                   ;512 BYTES BIS ENDE
LD HL,(BUFFER)                    ;AB HIER AUSGEBEN
CALL SEMO                         ;MODE SETZEN
AUSBY:LD A,H                      ;HIGH BYTE
CALL HEXAUS                       ;AUSGEBEN
LD A,L                            ;LOW BYTE
CALL HEXAUS                       ;AUSGEBEN
LD A,&20                          ;[SPACE]
CALL &BB5A                        ;CHAR OUT
LD A,(MOFLAG)                     ;A=LAENGE
LD B,A                            ;16 ZEICHEN
AUSAKKU:LD A,(HL)                 ;BYTE IN AKKU
CALL HEXAUS                       ;AUSGEBEN
LD A,&20                          ;[SPACE]
CALL &BB5A                        ;AUSGEBEN
INC HL                            ;HL = HL + 1
DJNZ AUSAKKU                      ;SCHLEIFE UNTIL 16
LD A,&20                          ;[SPACE]
CALL &BB5A                        ;AUSGEBEN
LD DE,(MOFLAG)                    ;16 TIMES
OR A                              ;DELETE CARRY
SBC HL,DE                         ;POINTER - 16
LD A,(MOFLAG)                     ;MODE ?
LD B,A                            ;POINTER
AUSLOOP:LD A,(HL)                 ;BYTE IN AKKU
INC HL                            ;HL = HL + 1
RES 7,A                           ;DELETE BIT 7
CP &20                            ;ASCII GROESSER 32 ?
JR NC,PRAUS                       ;JAWOHL, AUSGEBEN
```

Drucker. Beim NLQ 401 muß der DIP-Schalter für AUTO LINE FEED auf ON stehen, da sonst kein Zeilenvorschub ausgeführt wird.
Beispiel:
Das Directory der Usernummer 3 soll auf Drucker ausgegeben werden. Sie geben ein:
IUSER,3
IPCAT
IUSER,0

12. BHELP
SYNTAX :IBHELP
FUNKTION:
Dieser Befehl ist eine reine Hilfsroutine für das Disc-Basic V3.0 und gibt eine Liste sämtlicher Befehle und Parameterübergaben auf dem Bildschirm aus. Dies funktioniert in MODE 1 und MODE 2. In MODE 0 sollte dieser Befehl nicht angewendet werden, weil alles ein wenig unübersichtlich aussieht.

13. INIT
SYNTAX :IINIT
FUNKTION:
Der Befehl IINIT bewirkt eine komplette Initialisierung der Text-VDU des Screen- und Keyboard-Managers. Eine Anwendung findet sich darin, daß bei veränderten Farben und Zeichensatz wieder der Normalzustand des Rechners hergestellt werden kann. Empfehlenswert ist es, diesen Befehl mit
KEY 128,"IINIT"+CHR$(13)
auf die 0 des Zehnerblocks zu legen.

14. WAIT
SYNTAX :IWAIT
FUNKTION:
Der Befehl IWAIT unterbricht den Programmablauf solange, bis irgendeine Taste gedrückt wird.

15. AKKU
SYNTAX :IAKKU,BYTE
FUNKTION:
Das Register A wird mit dem Wert (BYTE) geladen und in einem temporären Speicher abgelegt. Siehe hierzu auch den Befehl ISYS.

16. HL
SYNTAX :IHL,BYTE1, BYTE2
FUNKTION:
Das Register H wird mit dem Wert (BYTE1) geladen, das Register L mit dem Wert (BYTE2). Beide Werte werden in einem TEMP abgelegt. Siehe hierzu auch den Befehl ISYS.
17. BC

```
LD A,46                              ;[.]
PRAUS:CALL &BB5A                     ;AUSGEBEN
DJNZ AUSLOOP                         ;SCHLEIFE UNTIL 16
LD A,&0D                             ;CR
CALL &BB5A                           ;AUSGEBEN
LD A,&0A                             ;LF
CALL &BB5A                           ;AUSGEBEN
PUSH HL                              ;HL AUF STACK RETTEN
LD DE,(SCHLUSS)                      ;ENDE FESTSTELLEN
OR A                                 ;DELETE CARRY
SBC HL,DE                            ;HL - ENDE
POP HL                               ;HL VOM STACK
JR C,AUSBY                           ;KLEINER 0 ?? CONTINUE
JR Z,AUSBY                           ;GLEICH  0 ?? CONTINUE
LD A,(PFLAG)                         ;STREAMFLAG HOLEN
CP 1                                 ;<> 1
CALL Z,SWAPOFF                       ;VEKTOR RESTAURIEREN
RET                                  ;ZURUECK
PFLAG: DEFB 0                        ;1 BYTE FLAG
SCHLUSS: DEFS 2                      ;2 BYTE FLAG
;---------------------------------   *** SUB CHAR TO HEX ***
HEXAUS: LD E,A                       ;AKKU MERKEN
RRCA                                 ;FOUR TIMES
RRCA                                 ;ROTATE
RRCA                                 ;RIGHT
RRCA                                 ;AKKU
AND %1111                            ;DELETE BIT 4-7
CALL NIBBINV                         ;NIBBLE AUSGEBEN
LD A,E                               ;AKKU ZURUECK
AND %1111                            ;DELETE BIT 4-7
CALL NIBBINV                         ;NIBBLE AUSGEBEN
RET                                  ;ZURUECK
;---------------------------------   *** SUB UMWANDLER ***
NIBBINV: CP &0A                      ;<10 ODER WAS
JR C,WERT                            ;MAYBE EIN WERT
ADD A,7                              ;ODER EIN ZEICHEN ?
WERT:ADD A,48                        ;ASCII
CALL &BB5A                           ;AUSGEBEN
RET                                  ;ZURUECK
;---------------------------------   *** SUB SWAP STREAMS ***
SWAP:LD A,&01                        ;FLAG SET
LD (PFLAG),A                         ;FLAG EINRICHTEN
LD A,&C3                             ;-FIRMWARE
LD (&BB5A),A                         ;-VEKTOR
LD A,&2B                             ;-BILDSCHIRMAUSGABE
LD (&BB5B),A                         ;-AUF
LD A,&BD                             ;-DRUCKER
LD (&BB5C),A                         ;-VERBIEGEN
RET                                  ;ZURUECK
;---------------------------------   *** SUB SWAP STREAMS OFF ***
SWAPOFF: LD A,&00                    ;FLAG RESET
LD (PFLAG),A                         ;FLAG LOESCHEN
LD A,&CF                             ;-FIRMWARE
LD (&BB5A),A                         ;-VEKTOR
LD A,&00                             ;-BILDSCHIRMAUSGABE
LD (&BB5B),A                         ;-WIEDER AUF
LD A,&94                             ;-BILDSCHIRM
LD (&BB5C),A                         ;-EINRICHTEN
RET                                  ;-ZURUECK
;---------------------------------   *** SUB FEHLERMELDUNG ***
FEHLER:LD HL,FTEXT                   ;TEXT NACH HL
;---------------------------------   *** SUB AUSGABE VON TEXTEN ***
TXTOUT:LD A,(HL)                     ;A IN HL
CP &00                               ;TEXTENDE ?
RET Z                                ;ZURUECK IF TRUE
CALL &BB5A                           ;AUSGEBEN
INC HL                               ;HL = HL + 1
JP TXTOUT                            ;SCHLEIFE WEITER
;---------------------------------   *** SUB FEHLERTEXT ***
FTEXT:DEFB &0A,&0D                   ;LF/CR
DEFM "!!! PARAMETERFEHLER !!!"       ;FEHLERTEXT
DEFB &07,&0A,&0D,&00                 ;KLINGEL/LF/CR ENDE TABELLE
;---------------------------------   *** SUB CLEAN ***
CLEAN:LD A,1                         ;1 PARAMETER
LD IX,JOKER                          ;ADRESSE NAME
RST &18,ERACAL                       ;ERASE FILE
RET                                  ;ZURUECK
ERACAL:DEFB &2D,&C0,&07              ;ROM 7 ADRESSE &C02D
JOKER:DEFW JOKER1                    ;NAME
JOKER1:DEFB 5                        ;5 ZEICHEN
DEFW NAME                            ;POINTER
NAME:DEFM "*.BAK"                    ;WILDCARD + NAME
;---------------------------------   *** SUB WAIT ***
WARTE:CALL &BB06                     ;WAIT ON KEY
RET                                  ;ZURUECK
;---------------------------------   *** SUB INIT ***
REDEF:CALL &BB00                     ;KEYBOARD INIT
CALL &BB4E                           ;TXT INIT
CALL &BBBA                           ;GRA INIT
CALL &BBBF                           ;SCR INIT
RET                                  ;ZURUECK
;---------------------------------   *** SUB MINI MONITOR ***
```

```
MINIMON:CP 3                     ;DREI PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD A,&0C                         ;CHAR FOR CLS
CALL &BB5A                       ;DO THE CLS
LD A,(IX)                        ;DRITTER PARAMETER [STREAMNUMMER]
CP 1                             ;<> 1
CALL Z,SWAP                      ;VEKTOR EINRICHTEN
LD D,(IX+3)                      ;BYTE 1
LD E,(IX+2)                      ;ENDE
LD (SCHLUSS),DE                  ;ZWEITER PARAMETER [ENDE]
LD H,(IX+5)                      ;BYTE 1
LD L,(IX+4)                      ;ERSTER PARAMETER  [START]
CALL SEMO                        ;MODE SETZEN
CALL AUSBY                       ;SPRUNG IN AUSGABEROUTINE
RET                              ;FERTIG
;-------------------------------- *** SUB MODE SCANNEN UND SETZEN ***
SEMO:CALL &BC11                  ;MOMENTANER MODE NACH A
CP 2                             ;MODUS 2 ?
JP Z,SEMO2                       ;JA, AUF 2 SETZEN
CP 1                             ;MODUS 1 ?
JP Z,SEMO1                       ;JA, AUF 1 SETZEN
LD A,&03                         ;3 ZEICHEN
LD (MOFLAG),A                    ;MODE 0, LAENGE = 3
JP BACK                          ;GESETZT, ZURUECK
SEMO1:LD A,&08                   ;8 ZEICHEN
LD (MOFLAG),A                    ;MODE 1, LAENGE = 8
JP BACK                          ;GESETZT, ZURUECK
SEMO2:LD A,&0F                   ;16 ZEICHEN
LD (MOFLAG),A                    ;MODE 2, LAENGE = 16
BACK:RET                         ;ZURUECK
MOFLAG:DEFS 2                    ;2 BYTE SPEICHER
;-------------------------------- *** SUB HL ***
HREG:CP 2                        ;2 PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD A,(IX+2)                      ;LOW BYTE
LD (HFLAG),A                     ;ERSTER PARAMETER
LD A,(IX)                        ;HIGH BYTE
LD (LFLAG),A                     ;ZWEITER PARAMETER
RET                              ;ZURUECK
;-------------------------------- *** SUB DE ***
DREG:CP 2                        ;2 PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD A,(IX+2)                      ;LOW BYTE
LD (DFLAG),A                     ;ERSTER PARAMETER
LD A,(IX)                        ;HIGH BYTE
LD (EFLAG),A                     ;ZWEITER PARAMETER
RET                              ;ZURUECK
;-------------------------------- *** SUB BC ***
BREG:CP 2                        ;2 PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD A,(IX+2)                      ;LOW BYTE
LD (BFLAG),A                     ;ERSTER PARAMETER
LD A,(IX)                        ;HIGH BYTE
LD (CFLAG),A                     ;ZWEITER PARAMETER
RET                              ;ZURUECK
;-------------------------------- *** SUB AF ***
AREG:CP 1                        ;1 PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD A,(IX)                        ;LOW BYTE
LD (AFLAG),A                     ;PARAMETER
RET                              ;ZURUECK
;-------------------------------- *** SUB REGFLAGS ***
HFLAG:DB 00                      ;H REGISTER
LFLAG:DB 00                      ;L REGISTER
DFLAG:DB 00                      ;D REGISTER
EFLAG:DB 00                      ;E REGISTER
BFLAG:DB 00                      ;B REGISTER
CFLAG:DB 00                      ;C REGISTER
AFLAG:DB 00                      ;A REGISTER
;-------------------------------- *** SUB SYSCALL ***
SYSCA:CP 1                       ;1 PARAMETER ?
JP NZ,FEHLER                     ;PARAMETERFEHLER IF NOT
LD D,(IX+1)                      ;BYTE 1 ADRESSE NACH D
LD E,(IX)                        ;BYTE 2 ADRESSE NACH E
LD (SYSFLAG),DE                  ;ADRESSE MERKEN
LD A,(HFLAG)                     ;REGISTER MIT DEFAULTS LADEN
LD H,A                           ;-
LD A,(LFLAG)                     ;-
LD L,A                           ;-
LD A,(DFLAG)                     ;-
LD D,A                           ;-
LD A,(EFLAG)                     ;-
LD E,A                           ;-
LD A,(BFLAG)                     ;-
LD B,A                           ;-
LD A,(CFLAG)                     ;-
LD C,A                           ;-
LD A,(AFLAG)                     ;-
LD IX,(SYSFLAG)                  ;ADRESSE NACH IX
JP (IX)                          ;AUFRUF
RET                              ;ZURUECK
SYSFLAG:DEFB &01,&AC             ;ADRESSE DEFAULT &AC01 [RET]
```

```
1 '************************
2 '*    SEKTOR MAP V3.0    *          [783]
3 '*----------------------*           [1417]
4 '* CPC INTERNATIONAL 86 *           [1693]
5 '* -------------------- *           [1284]
6 '* THOMAS MORGEN 1986   *           [785]
7 '************************           [768]
8 '                                   [117]
9 '                                   [117]
10 MEMORY &4FFF:GOSUB 170:CALL &5000  [1025]
20 MODE 2:fflag =&505B:PRINT"       SEKTO [13501]
R-MAP V1.0 BY TM.....INSERT DISC THEN PRES
S ANY KEY":PRINT"------------------------
------------------------------------------
------------":CALL &BB06
30 DIM F(7,8),N(7)                  [875]
40 FOR G = 0 TO 4                   [663]
50 FOR T = 0 TO 7:TRACK = G*8+T:GO=1 [1386]
60 |SECSCAN,TRACK:IF PEEK (FFLAG) <>0 THEN [6432]
 GO = GO +1:IF GO<>101 THEN 60 ELSE N(T)=0
:GOTO 120
70 MI=256:BUFF=&505C:XBUFF=BUFF     [2372]
80 P=PEEK(BUFF):IF P=MI THEN 110    [1933]
90 IF P<MI THEN MI=P:XBUFF=BUFF     [2382]
100 BUFF=BUFF+1:GOTO 80             [1268]
110 N(T)=BUFF-XBUFF:IF N(T)>9 OR (N(T)<8 A [5582]
ND N(T)<>0) THEN N(T)=9
120 PRINT CHR$(24);"TRACK:";CHR$(24);TRACK [4119]
;"  ";:IF TRACK<10 THEN PRINT" ";
130 IF N(T)=0 THEN PRINT"     NICHT FORMAT [2219]
IERT      ";
140 FOR S= 0 TO N(T)-1:F(T,S)=PEEK(S+XBUFF [3733]
):PRINT HEX$(F(T,S),2);" ";
150 NEXT:NEXT:NEXT                  [963]
160 END                             [110]
170 DATA &01,&0D,&50,&21,&09,&50,&C3,&D1,& [4191]
BC,&00,&00,&00,&00,&12,&50,&C3,&1A,&50,&53
,&45, 1359
180 DATA &43,&53,&43,&41,&CE,&00,&FE,&01,& [3426]
C0,&11,&FF,&01,&CD,&46,&50,&1E,&00,&DD,&56
,&00, 1900
190 DATA &DF,&4F,&50,&21,&5C,&50,&E5,&DF,& [3783]
52,&50,&E1,&30,&0B,&3A,&51,&BE,&77,&23,&7D
,&FE, 2347
200 DATA &1F,&20,&EF,&AF,&32,&5B,&50,&11,& [3475]
00,&0A,&7B,&DF,&55,&50,&7A,&DF,&58,&50,&C9
,&63, 2049
210 DATA &C7,&07,&6C,&C5,&07,&72,&CA,&07,& [2480]
03,&C6,&07, 1049
220 dat=0 : sz=0 : dz = 170         [1743]
230 FOR adr = 20480  TO 20570       [841]
240   READ byte : dat=dat+1         [1921]
250   sz=sz+byte                    [1743]
260   POKE adr,byte                 [898]
270   IF dat < 20 AND adr <  20570 THEN 31 [1881]
0
280   READ chksum                   [1079]
290   IF chksum<>sz THEN PRINT "Fehler in [3061]
Zeile :";dz
300   dz=dz + 10 : sz=0 : dat=0     [2078]
310 NEXT adr                        [453]
320 RETURN                          [555]
```

SYNTAX :|BC,BYTE1, BYTE2
FUNKTION:
Das Register B wird mit dem Wert (BYTE1) geladen, das Register C mit dem Wert (BYTE2). Beide Werte werden in einem TEMP abgelegt. Siehe hierzu auch den Befehl |SYS.

18. DE
SYNTAX :|BC,BYTE1, BYTE2
FUNKTION:
Das Register D wird mit dem Wert (BYTE1) geladen, das Register E mit dem Wert (BYTE2). Beide Werte werden in einem TEMP abgelegt. Siehe hierzu auch den Befehl |SYS.

19. SYS
SYNTAX :|SYS,ADRESSE
FUNKTION:
Alle Register werden mit den, über die Befehle AKKU, HL,BC, DE gefüllten TEMPS geladen und danach die Adressen angesprungen. Dadurch ist es möglich, ohne Einladen eines Assemblers auf einfache Weise Werte für Firmwareroutinen zu übergeben.

20. HDUMP
SYNTAX :|HDUMP,START, ENDE,DEVICE
FUNKTION:
Der Befehl HDUMP arbeitet ähnlich wie der Befehl EXAMINE. Hier lassen sich jedoch Start- und Endadresse festlegen und so ein Hex-ASCII-Dump eines Speicherbereiches ausgeben. Die Parameter für Device werden genau wie bei EXAMINE

```
;*********************
;*    SEKTOR SCANNER  *
;*-------------------*
;*    COPYRIGHT 1986  *
;* CPC INTERNATIONAL  *
;*-------------------*
;* THOMAS MORGEN  86  *
;*********************
;
ORG &5000                        ;STARDTADRESSE
LD BC,RSX                        ;RSX UEBERNEHMEN
LD HL,KERNAL                     ;BUFFER UEBERNEHMEN
JP &BCD1                         ;UND EINBINDEN
KERNAL:DEFS 4                    ;BUFFER 4 BYTES
RSX:DEFW TABLE                   ;SPRUNGTABELLE
JP SCAN                          ;BEFEHL 1
TABLE:DEFM "SECSCA"              ;BEFEHLSWORTE
DEFB "N"+&80                     ;-
NOP                              ;ENDE TABELLE
;------------------------------- *** SUB SEKTOR SCANNEN ***
SCAN:CP 1                        ;1 PARAMETER ?
RET NZ                           ;ZURUECK IF NOT
LD DE,&01FF                      ;DEFAULT DRIVE PRAMETER
CALL SEDRV                       ;INSTALLIEREN
LD E,00                          ;E RESETEN
LD D,(IX)                        ;TRACK PARAMETER
RST &18,LOCATE                   ;DRIVE HEAD ON TRACK
LD HL,OFFSETS                    ;IN BUFFER
SLOOP:PUSH HL                    ;HL AUF STACK
RST &18,SCANSEC                  ;SCANNEN
POP HL                           ;HL VOM STACK
JR NC,NOFORM                     ;NICHT FORMATIERT
LD A,(&BE51)                     ;LAST OFFSET
LD (HL),A                        ;IN BUFFER
INC HL                           ;BUFFER ERHOEHEN
LD A,L                           ;CHECK END
CP &1F                           ;= &1F ?
JR NZ,SLOOP                      ;SCHLEIFE IF NOT
XOR A                            ;A RESETEN
NOFORM:LD (FORMFLAG),A           ;ERROR
LD DE,&0A00                      ;DEFAULT DRIVE PARAMETER
SEDRV:LD A,E                     ;FEHLERMELDUNGEN
RST &18,EROFF                    ;ON/OFF
LD A,D                           ;LESEVERSUCHE
RST &18,SCTRY                    ;SETZEN
RET                              ;ZURUECK
;------------------------------- *** SUB TEMPS/FLAGS/DEFAULTS ***
LOCATE:DEFB &63,&C7,&07          ;ROM 7,ADRESSE &C763
SCANSEC:DEFB &6C,&C5,&07         ;ROM 7,ADRESSE &C56C
EROFF:DEFB &72,&CA,&07           ;ROM 7,ADRESSE &CA72
SCTRY:DEFB &03,&C6,&07           ;ROM 7;ADRESSE &C603
FORMFLAG:DEFB 00                 ;ERRORFLAG
OFFSETS:DEFS 9                   ;TEMP OFFSETS
```

DISC-BASIC V3.0 REFERENZ

Folgende Referenzen zu den Disc-Basic-Routinen können vom Anwender auf MC-Ebene genutzt werden. Diese Einsprünge stellen eine Art Firmware dar und sollen dem User helfen, mehr aus dem Disc-Basic zu machen.

ADRESSE	BEMERKUNG	EINSPRUNG
&8000	STARTADRESSE	--------
&80A6	BEFEHL LIES SEKTOR	IX+4=DRIVE IX+2=TRACK IX=SEKTOR
&80BE	BEFEHL SCHREIB SEKTOR	IX+4=DRIVE IX+2=TRACK IX=SEKTOR
&80D6	BEFEHL BUFFER SETZEN	IX=BUFFERSTART
&80E6	ZEIGER AUF BUFFER	--------
&80E8	TURBO DOS ON	--------
&80F2	TABELLE FUER TURBO DOS	--------
&80FB	TURBO DOS OFF	--------
&8102	TABELLE NORMAL	--------
&810B	BEFEHL DELCAT	--------
&8121	BEFEHL PCAT	--------
&812E	BEFEHL FORMATEX	IX+4=DRIVE IX+2=TRACK IX=OFFSET
&8180	SELEKTIERTER OFFSET	--------
&8181	TABELLE FUER FORMATEX	--------
&81A7	BEFEHL FORMATIERE	IX=SELEKTION
&8200	TABELLE WARTETEXT	--------
&821E	FLAG DATA/VENDOR	--------
&821F	40 SPUREN BEI FORMAT	--------
&8220	DRIVE BEIM FORMAT	--------
&8225	TABELLE VENDOR FORMAT	--------
&8249	TABELLE DATA FORMAT	--------
&826D	BEFEHL HELP	--------
&827C	TEXTTABELLE HELP	--------
&854D	BEFEHL EXAMINE	IX=AUSGABEKANAL
&85CB	AUSGABEKANAL FLAG	--------
&85CE	ZEICHEN IN HEX WANDELN	A=ZEICHEN
&85EB	SCREENVEKTOR AUF DRUCKER	--------
&8600	SCREENVEKTOR ZURUECK	--------
&8618	TEXT AUSGEBEN	HL=TEXTTABELLE [ENDE = 0]
&8623	TABELLE PARAMETERFEHLER	--------
&8640	BEFEHL CLEAN	--------
&8652	NAME FUER CLEAN [4 ZEICHEN *.BAK]	--------
&8657	BEFEHL WAIT	--------
&865B	BEFEHL INIT	--------
&8668	BEFEHL HDUMP	IX+4=ENDE IX+2=START IX=AUSGABEKANAL
&8691	MODUS FESTSTELLEN	--------
&86B4	FLAG FUER MOMENTANEN MODE	--------
&86B6	BEFEHL HL	IX+2=H IX=L
&86C8	BEFEHL DE	IX+2=D IX=E
&86DA	BEFEHL BC	IX+2=B IX=C
&86EC	BEFEHL AKKU	IX=A
&86F8	TEMP FUER H	--------
&86F9	TEMP FUER L	--------
&86FA	TEMP FUER D	--------
&86FB	TEMP FUER E	--------
&86FC	TEMP FUER B	--------
&86FD	TEMP FUER C	--------
&86FE	TEMP FUER A	--------
&86FF	BEFEHL SYS	PARAMETER IN TEMPS

übergeben. Auch für die Bildschirmmodi gilt hier das gleiche. Eine Unterbrechung des Ausdrucks ist nicht möglich.

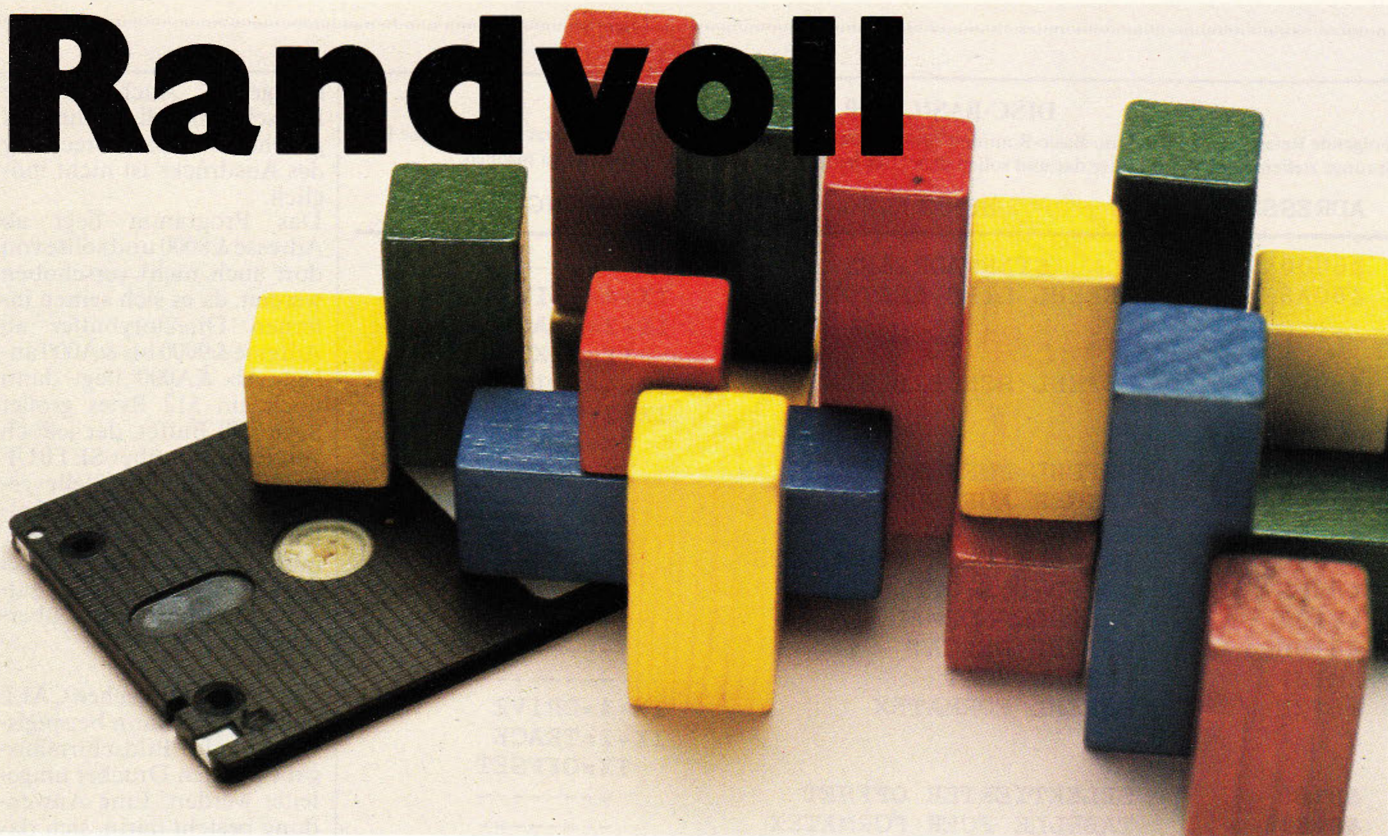
Das Programm liegt ab Adresse &8000 und sollte von dort auch nicht verschoben werden, da es sich seinen internen Directorybuffer ab Adresse &9000 bis &A000 anlegt. Ab &A000 liegt dann noch ein 512 Bytes großer Sektor I0 Buffer, der jedoch von Basic aus über SETBUF an eine beliebige Stelle gelegt werden kann. Wie bei allen verschobenen Buffern muß jedoch darauf geachtet werden, daß er das eigentliche Disc-Basic nicht überschreibt.

Durch einen einfachen CALL nach &85EB kann beispielsweise jede Bildschirmausgabe auf den Drucker umgeleitet werden. Eine Anwendung besteht darin, sich das Directory der gelöschten Files auf Drucker ausgeben zu lassen, indem Sie eintippen:

CALL &85EB
IDELCAT
CALL &8600

Als letztes Tool zum Disc-Basic gibt es noch das Programm SEKTOR MAP, welches einen weiteren Basic-Befehl zur Verfügung stellt, der jedoch nur Anwendung in diesem Hilfsprogramm findet und deswegen nicht in Disc-Basic mit aufgenommen wurde. SEKTOR MAP erstellt ein Inhaltsverzeichnis der Spuren einer Disk und gibt an, ob ein Track nicht formatiert wurde. Da sich eine nicht formatierte Spur schlecht lesen läßt und dies nur über Manipulationen der Fehlermeldungen und der Anzahl der Leseversuche verwirklichen ließe, ist diese Routine nicht gerade schnell. Jemand, der seine Disketten jedoch mit FORMATEX behandelt hat, wird rasch den praktischen Wert des Programmes erkennen. SEKTOR MAP liegt ab &5000 und kann sich gleichzeitig mit dem Disc-Basic im Speicher befinden, ohne sich gegenseitig zu behindern. Bei einigen 664 und 6128 Versionen kann es durch geänderte Ausgaberoutinen zu leichten Schwierigkeiten kommen. Hängen Sie in dem Fall einfach hinter die erste FOR-NEXT-Schleife noch ein CHR$(13). Das Programm arbeitet zwar einwandfrei, der Bildschirmaufbau ist jedoch etwas verschoben. (TM)

Randvoll



Auf den Disketten im original Schneider-Format ist der vorhandene Speicherplatz nicht gerade üppig bemessen. 180K-Byte – großzügig gerechnet – sind speziell, wenn man mit sehr umfangreichen Programmen und Datensammlungen arbeitet, einfach arg knapp. Deshalb sind Schneider-Freaks immer auf der Suche nach speicherplatzsparenden Routinen und Möglichkeiten, ihr vorhandenes System möglichst effektiv nutzen zu können. Beim Analysieren des Diskettenbetriebssystems und seiner Möglichkeiten kam mir eine Idee, die ich an dieser Stelle einmal weitergeben möchte. Bei Leuten, die sich ebenfalls mit diesem Thema auseinander gesetzt haben, ist bereits seit längerer Zeit bekannt, daß die 3"-Diskettenlaufwerke sich bis zur 43. Spur formatieren lassen.

Das bringt rein theoretisch einen Gewinn von ca. 13 weiteren KByte, auf denen dann Daten aufgezeichnet werden könnten. Leider setzt das AMSDOS dieser Nutzung jedoch Widerstand entgegen, da es für ein Lesen dieser Spuren nicht eingerichtet ist. Diese Spuren lassen sich nur mit Hilfe speziell dafür ausgelegter Programme lesen und benutzen. Eine Ausnahme gibt es jedoch, und die soll hier dargestellt werden. Programme, die auf der 41. oder der 43. Spur abgelegt werden, lassen sich z.B. von Basic aufrufen. Sind sie jedoch länger als acht Sektoren, treten Fehlermeldungen auf, deshalb sollte man sich auf Programme beschränken, die nicht mehr als 4K-Byte Umfang haben. Ein insgesamt 8K-Byte Programm ist schon eine ganze Menge, speziell wenn es sich um ein in Maschinensprache geschriebenes Programm handelt, aber auch in Basic kann man hier schon eine ganze Menge unterbringen. So z.B. eine kleine Textverarbeitung, einen Disc-Monitor, Maschinensprachemonitore und vieles mehr. Auch ist, was den Umfang der Programme angeht, sicherlich noch nicht das letzte Wort gesprochen. Es werden sich bestimmte Freaks finden, die hier weitergehende Möglichkeiten finden oder gar schon gefunden haben, um Programme auch über die genannten Spuren hinaus ohne Fehlermeldung des Systems aufrufen zu können.

Wie gelangt man jetzt in den Genuß dieser zusätzlichen KBytes? Zunächst einmal benötigt man dazu etwas Handwerkszeug, in unserem Falle ein Programm, das es erlaubt, die 41. und 43. Spur auf der Diskette zu formatieren. Außerdem muß man in der Lage sein, Sektoren auf der Diskette einzeln zu lesen, sie zu bearbeiten und anschließend wieder an der gleichen oder einer anderen Stelle der Diskette abspeichern zu können. Viele der käuflich zu erwerbenden Diskettenmonitore und auch einige in Computerzeitschriften vorgestellte, gestatten dies bereits oder sind nach Umschreiben des Programmes dazu in der Lage. Um das gewünschte Programm auf die vorgesehene Spur zu bekommen, geht man am besten folgendermaßen vor:

Zunächst nimmt man eine unbeschriebene Diskette und formatiert sie im Data-Format bis zur 43. Spur. Anschließend wird das Programm, das dort abgelegt werden soll, auf diese Diskette abgespeichert. Nun liest man das Directory dieser Diskette ein. Da sich nur dieses eine Programm auf der Diskette befindet, ist es mit Sicherheit auf Spur 0 im Sektor 1 zu finden. In den ersten 16 Bytes des Directoryeintrages finden wir der Reihe nach aufgeführt:

1.	Byte:	Usernummer (hex 0–15)
2. – 12.	Byte:	Filename mit Extension
16. –	Byte:	Filegröße (X 128 = Größe des Files in Byte)
17. – 32.	Byte:	Position des Files auf der Diskette

Anhand des 17. – 20. Byte bei einem 4K-Programm (1 Byte pro belegtes KByte auf der Diskette) können wir nun die Lage des Programmes auf der Diskette ermitteln. Hex 02 entspricht Spur 0, Sektor 5, hex 03 Spur 0. Sektor 7, hex 04 Spur 0, Sektor 9 usw. Um nicht jedesmal rechnen zu müssen, sind weitere Umrechnungen in einer separaten Tabelle aufgeführt (siehe Tabelle 1). Es gilt nun, die Sektoren, die in dem 17. – 20. Byte des Directoryeintrages aufgeführt sind, sektorenweise auf die 41. Spur zu kopieren, ohne sie dabei verändern zu müssen. Dazu müssen diese Sektoren, in unserem Beispiel die Sektoren 5 – 9 auf Spur 0 und 1 – 3 auf Spur 1 der Reihe nach eingelesen und auf der Spur 40, in den Sektoren 1 – 8 direkt anschließend wieder abgespeichert werden. Damit ist die Übertragung des Programmes bereits geschehen und es muß nur noch dafür gesorgt werden, daß unser Programm auch vom Computer wiedergefunden wird. Dazu sind die Bytes 17 – 20 des Directoryeintrages in die Werte umzuändern, die der Position des Programmes auf Spur 40 entsprechen. Diese Werte sind wieder der Tabelle zu entnehmen. Anschließend kann das Programm, wie gewohnt mit <RUN >"Programmname" gestartet werden. Wenn man sich jedoch jetzt mit dem "CAT"-Befehl ansieht, wieviel Platz noch auf der Diskette vorhanden ist, wird man feststellen, daß das Programm auf der 41. Spur scheinbar keinen Platz belegt, es werden 178K Free angezeigt. Das ursprüngliche Programm ist zwar auch noch auf der Diskette vorhanden, wird aber von der Diskettenverwaltung nicht mehr zur Notitz genommen, da kein Zeiger des Directorys mehr darauf zeigt und der von dem Programm ursprünglich eingenommene Platz wird deshalb von der Diskettenverwaltung beim Bespielen der Diskette mit ande-

ren Programmen neu vergeben. Mit einem zweiten Programm, das auf der 43. Spur abgespeichert wird, Spur 42 wenn man mit Spur 0 zu zählen beginnt, wird anschließend ähnlich verfahren. Auch dieses Programm läßt sich, genauso wie das erste, mit <RUN> "Programmname" starten. Leider ließ sich aus Gründen, die der Autor bis jetzt noch nicht ermitteln konnte, die Spur 41 nicht in ähnlicher Weise verwenden. Das hier geschilderte Verfahren läßt sich auch mit Sicherheit noch verbessern. Leider fehlte bis jetzt noch die Zeit, die Methode etwas zu verfeinern und die Abläufe vom Computer steuern zu lassen. Ein passendes Computerprogramm kann deshalb leider noch nicht mitgeliefert werden.

Zum Schluß noch einige Hinweise, wie man das, auf diese Art und Weise, recht mühsam transferierte Programm relativ wirkungsvoll schützt. Durch Addieren einer hexadezimalen 80 auf den Hexwert des ersten Buchstabens der Dateikennung und Eintragen dieses neuen Wertes an der 10. Stelle des Directoryeintrages, wird unser transferiertes Programm zum Read-Only-File erklärt und läßt sich nicht mehr löschen, es sei denn, man formatiert die Diskette neu oder setzt den Wert im Directoryeintrag zurück. Das hier geschilderte Verfahren zum Übertragen von Programmen auf die 41. und 43. Spur einer Diskette läßt sich nur von Basic aus verwenden, unter CP/M funktioniert es nicht. Programme sollten übrigens auf eine solchermaßen vorbereitete Diskette nur mit <SAVE> abgespeichert werden, da bei Verwendung eines Kopierprogrammes wie z.B. »Disckit« sonst die Einträge im Directory wieder überschrieben werden. Man kann sich damit helfen, daß man anschließend mit Hilfe eines Discmonitors die Einträge von Hand wieder herstellt. Beim Kopieren ganzer Disketten ist dieses Verfahren sicher vorzuziehen, da sonst die Disketten laufend gewechselt werden müssen und man wesentlich mehr »Tipparbeit« leisten muß. »Disckit« überschreibt, das auf die oben geschilderte Art und Weise, auf die Spuren 40 und 42 geschriebene Programm übrigens nicht, es befindet sich auch nach dem Formatieren und Kopieren, mit Hilfe dieses Programmes, noch auf der Diskette! Nur die entsprechenden Directoryeinträge müssen eben wieder hergestellt werden.

W. Wantia

Tabelle zur Umrechnung Block (Directoryeintrag) in Spur und Sektor

Bl	Sp	Se	Bl	Sp	Se	Bl	Sp	Se	Bl	Sp	Se	Bl	Sp	Se	Bl	Sp	Se
02	0	5	27	8	7	4C	16	9	71	25	2	96	33	4	BB	41	6
03	0	7	28	8	9	4D	17	2	72	25	4	97	33	6	BC	41	8
04	0	9	29	9	2	4E	17	4	73	25	6	98	33	8	BD	42	1
05	1	2	2A	9	4	4F	17	6	74	25	8	99	34	1	BE	42	3
06	1	4	2B	9	6	50	17	8	75	26	1	9A	34	3	BF	42	5
07	1	6	2C	9	8	51	18	1	76	26	3	9B	34	5	C0	42	7
08	1	8	2D	10	1	52	18	3	77	26	5	9C	34	7	C1	42	9
09	2	1	2E	10	3	53	18	5	78	26	7	9D	34	9			
0A	2	3	2F	10	5	54	18	7	79	26	9	9E	35	2			
0B	2	5	30	10	7	55	18	9	7A	27	2	9F	35	4			
0C	2	7	31	10	9	56	19	2	7B	27	4	A0	35	6			
0D	2	9	32	11	2	57	19	4	7C	27	6	A1	35	8			
0E	3	2	33	11	4	58	19	6	7D	27	8	A2	36	1			
0F	3	4	34	11	6	59	19	8	7E	28	1	A3	36	3			
10	3	6	35	11	8	5A	20	1	7F	28	3	A4	36	5			
11	3	8	36	12	1	5B	20	3	80	28	5	A5	36	7			
12	4	1	37	12	3	5C	20	5	81	28	7	A6	36	9			
13	4	3	38	12	5	5D	20	7	82	28	9	A7	37	2			
14	4	5	39	12	7	5E	20	9	83	29	2	A8	37	4			
15	4	7	3A	12	9	5F	21	2	84	29	4	A9	37	6			
16	4	9	3B	13	2	60	21	4	85	29	6	AA	37	8			
17	5	2	3C	13	4	61	21	6	86	29	8	AB	38	1			
18	5	4	3D	13	6	62	21	8	87	30	1	AC	38	3			
19	5	6	3E	13	8	63	22	1	88	30	3	AD	38	5			
1A	5	8	3F	14	1	64	22	3	89	30	5	AE	38	7			
1B	6	1	40	14	3	65	22	5	8A	30	7	AF	38	9			
1C	6	3	41	14	5	66	22	7	8B	30	9	B0	39	2			
1D	6	5	42	14	7	67	22	9	8C	31	2	B1	39	4			
1E	6	7	43	14	9	68	23	2	8D	31	4	B2	39	6			
1F	6	9	44	15	2	69	23	4	8E	31	6	B3	39	8			
20	7	2	45	15	4	6A	23	6	8F	31	8	B4	40	1			
21	7	4	46	15	6	6B	23	8	90	32	1	B5	40	3			
22	7	6	47	15	8	6C	24	1	91	32	3	B6	40	5			
23	7	8	48	16	1	6D	24	3	92	32	5	B7	40	7			
24	8	1	49	16	3	6E	24	5	93	32	7	B8	40	9			
25	8	3	4A	16	5	6F	24	7	94	32	9	B9	41	2			
26	8	5	4B	16	7	70	24	9	95	33	2	BA	41	4			

Unter Basic werden die Sektoren 1 – 4, Spur 0 für Directoryeinträge benutzt. Im Systemformat befinden sich die Einträge auf Spur 2.

Super-Sort

für 464-664-6128



Der schnelle Tabellen-Sort für numerische Tabellen

Dieser Sort weicht von den bisherigen Arten von Sortierprogrammen erheblich ab. Es ist ein Sort mit Diskettenbetrieb. Der Dateiname ist frei wählbar. Nach Eingabe der Feldzahl pro Satz wird die DIM-Anweisung für den Sort bestimmt und als Zeile ausgegeben. Die nächsten zwei Zeilen zeigen die Zeit für die Laderoutine der Tabelle. Die folgende Zeile gibt die Anzahl der geladenen Sätze an.

Nach der Eingabe der Feldnummer, nach der die Tabelle sortiert werden soll, erfolgt eine Prüfung, ob die Tabelle schon sortiert ist.

Ist die Tabelle sortiert, so wird der Sort beendet, ein Hinweis und die Zeit ausgegeben. Eine Kontrolle der Tabelle kann angewählt werden. Ist die Tabelle noch nicht sortiert, so beginnt der Sort und gibt entsprechende Kontrollzeilen aus.

Start Sort
Ende Satz.Nr.
Ende Reorg
Datei steht sortiert auf DISC
Sort ist beendet!
Danach kann wieder die Kontrolle gewählt werden.

Hinweis:
Die Tabelle wird automatisch begrenzt. Ist eine Datei größer als Daten geladen werden können, wird der Ladevorgang beendet. Dieser Teil der Tabelle kann trotzdem sortiert werden. Die maximal ladbaren Sätze mit den dazugehörigen Feldern werden zu Beginn des Sorts ausgewiesen:
"DIM a (xxx,xx) eingerichtet."
Der Monitor informiert vom Beginn an bis zum Schluß über den Ablauf des Programms.

Georg Brinkmann




```
10 MODE 1:BORDER 0:INK 0,0:INK 1,8:INK 2,2 [2213]
4:INK 3,11
20 PEN 2:PRINT"            S U P E R -- S O [1602]
R T"
30 PRINT"         fuer numerische Tabellen  [3326]
40 PEN 1:PRINT:PRINT"                   von [2229]
50 PRINT"              Georg Brinkmann      [2033]
60 PRINT"       "; STRING$(27,"")          [1532]
70 LOCATE 13,7:PEN 2:INPUT"Dateiname  ";na [2992]
$:PEN 1
80 LOCATE 6,8:INPUT"Feldzahl pro Satz eing [4039]
eben";fd
90 satz=INT(6280/fd):IF satz>1500 THEN sat [4111]
z=1000
100 DIM a(satz,fd):as=LEN(STR$(satz)):ds=L [3497]
EN(STR$(fd))
110 satz$=RIGHT$(STR$(satz),as-1)+CHR$(44) [3164]
+RIGHT$(STR$(fd),ds-1)
120 LOCATE 7,9:PRINT"DIM  ";:PEN 2:PRINT"a [6546]
(";satz$;")";:PEN 1:PRINT" eingerichtet"
130 INK 1,5:INK 2,7:INK 3,24                [1262]
140 d=INT(TIME/300):GOSUB 610               [945]
150 LOCATE 7,10:PRINT"Zeit  ";USING"## ";h [2125]
;m;s;:PRINT" Start Laden"
160 GOSUB 620:GOSUB 610                     [1113]
170 LOCATE 7,11:PRINT"Zeit  ";USING"## ";h [4326]
;m;s;:PRINT" Ende  Laden"
180 LOCATE 2,12:PEN 2:PRINT USING"### ";n; [7832]
:PRINT"Saetze mit je";USING" ### ";fd;:PRI
NT"Feldern geladen":PEN 1
190 PRINT""; STRING$(40,".");               [1194]
200 LOCATE 11,15:PEN 3:PRINT"Sortfield 1 - [5204]
";fd;:INPUT;q:PEN 1:u=n
210 FOR z=2 TO n                            [1165]
220 IF a(z,q) < a(z-1,q) THEN GOTO 290      [2069]
230 NEXT:LOCATE 12,17:PEN 3:PRINT"Datei wa [5182]
r sortiert"
240 PRINT STRING$(40,"_");:PEN 3            [1600]
250 GOSUB 610                               [923]
260 LOCATE 7,19:PRINT"
Zeit  ";USING"## "; [4740]
h;m;s;:PRINT"  Sort beendet"
270 INPUT"
Datei anzeigen   j/n";zeig$:IF  [4750]
zeig$="j" OR zeig$="J" GOTO 580
280 GOTO 600                                [413]
290 d=INT(TIME/300)                         [615]
300 GOSUB 610                               [923]
310 LOCATE 7,16:PRINT"Zeit  ";USING"## ";h [4300]
;m;s;:PRINT" Start Sort"
320 OPENOUT na$:PRINT#9,n : u=n             [2406]
330 u=u-1:IF u=0 GOTO 430                   [2861]
340 FOR z=1 TO n:IF a(z,0)=0 THEN su=z:su1 [3052]
=z:zs=1:GOTO 360
350 NEXT                                    [350]
360 x=z:IF zs=0 GOTO 430                    [1308]
370 FOR z=x TO n:IF a(z,0) > 0 GOTO 400     [1539]
380 IF a(z,q) < a(su,q)  THEN su=z          [3628]
390 IF a(z,q) > a(su1,q) THEN su1=z         [2295]
400 NEXT                                    [350]
410 FOR i=1 TO fd:PRINT#9,a(su,i):NEXT:FOR [4045]
 i=1 TO fd:PRINT#9,a(su1,i):NEXT
420 zs=0: a(su,0)=n-u: a(su1,0)=u+1:GOTO 3 [2898]
30
430 GOSUB 610                               [923]
440 LOCATE 7,17:PRINT"Zeit  ";USING"## ";h [4331]
;m;s;:PRINT" Ende  Satz.Nr."
450 CLOSEOUT                                [902]
460 GOSUB 620                               [921]
470 OPENOUT na$:PRINT#9,n                   [1855]
480 z=1                                     [470]
490 FOR x=1 TO fd:PRINT#9,a(z,x):NEXT:z=z+ [2180]
2:IF z > n GOTO 500 ELSE 490
500 IF n/2 = INT(n/2) THEN z=n ELSE z=n-1   [1613]
510 FOR x=1 TO fd:PRINT#9,a(z,x):NEXT:z=z- [2579]
2:IF z < 1 GOTO 520 ELSE 510
520 CLOSEOUT                                [902]
530 GOSUB 610                               [923]
540 LOCATE 7,18:PRINT"Zeit  ";USING"## ";h [4178]
;m;s;:PRINT" Ende  Reorg"
550 PEN 2:PRINT"
   Datei steht sortiert [2143]
 auf DISC"
560 PEN 3:PRINT"
          Sort ist beend [2911]
et!":PEN 3
570 INPUT"
      Datei anzeigen  j/n ";a [6626]
nz$:IF anz$="j" OR anz$="J" THEN GOSUB 620
:GOSUB 580:END ELSE END
580 MODE 2:INK 1,26:FOR z=1 TO n:FOR x=1 T [6157]
O fd:PRINT USING"####.## ";a(z,x);:NEXT:PR
INT:NEXT
590 PRINT:PRINT STRING$(24,"*");USING" ### [7626]
# ";z-1;:PRINT"Saetze  mit je";USING" ###
";fd;:PRINT"Felder  " STRING$(23,"*");
600 END                                     [110]
610 t=INT(INT(TIME/300)-d):h=INT(t/3600):m [5765]
=INT(t/60-(h*60)):s=t-((m*60)+(h*3600)):RE
TURN
620 OPENIN na$:LINE INPUT#9,n$:n=VAL(n$)    [2442]
630 IF n>satz THEN n=satz                   [2116]
640 FOR z=1 TO n:FOR x=1 TO fd:LINE INPUT# [3396]
9,a$
650 IF EOF=-1 THEN a(z,x)=VAL(a$):n=z:GOTO [2869]
 670
660 a(z,x)=VAL(a$):NEXT:NEXT                [2609]
670 CLOSEIN                                 [752]
680 RETURN                                  [555]
```

DISC-IN für 464



Mit dem Programm DISC-IN muß die Diskettenstation während längerer Tipparbeiten nicht ständig eingeschaltet sein, sondern kann später für die Abspeicherung aktiviert werden.

Um dies zu erreichen, geht man folgendermaßen vor:

1. Nachdem die Tipparbeit beendet ist, Floppy einschalten.

2. Umschalten auf Kassette mit ׀Tape

3. MERGE "DISC-IN"

4. RUN 60000

5. Abspeicherung des abgetippten Programms

Natürlich ist darauf zu achten, daß sich die Zeilennummern des abgetippten Programms und des Programms "DISC-IN" nicht überschneiden.

"DISC-IN" arbeitet auf folgende Weise:

1. Es wird die SYMBOL-Tabelle um den RAM-Bedarf der Floppy nach unten verschoben

2. Das Hintergrund-ROM Nr. 7 der Floppy wird initialisiert.

Gerhard Friedle

```
 60000 MEMORY &9FFF                          [134]
60001 DATA DD,6E,00,DD,66,01,ED,5B,96,B2,7 [2341]
2,11,00,05,ED,52,ED
60002 DATA 5B,94,B2,CD,AB,BB,21,FF,AB,01,0 [3427]
7,00,CD,CE,BC,C9
60003 s=0:RESTORE 60001:FOR i=&A000 TO &A0 [6507]
20:READ a$:a=VAL("&"+a$):POKE i,a:s=s+a:NE
XT
60004 IF s<>4250 THEN PRINT"Fehler in den  [2892]
Data-Zeilen!":END
60005 x%=0                                 [532]
60006 CALL &A000                           [637]
60007 DELETE 60000-60015:END               [1670]
60008 '******************************      [1285]
60009 '*           DISC-IN            *    [442]
60010 '*                              *    [175]
60011 '*     von: Gerhard Friedle     *    [1904]
60012 '*          A-5771 Leogang      *    [982]
60013 '*                              *
60014 '*                              *    [175]
60015 '******************************      [1285]
```

D-I-R für 464-664-6128



D-I-R steht für DISC-Information-Reader und ist ein Hilfsprogramm, das den Diskettenanwender auf einfache Weise hilft, Informationen über die einzelnen Files auf seine Diskette zu bekommen. Das Programm ist einteilig und sehr kompakt aufgebaut.

Leistungsmerkmale von D-I-R:
- ermittelt Startadresse von jedem Diskettenfile (ausgenommen CP/M-Files und Textdateien)
- ermittelt Länge in Bytes von jedem Diskettenfile (ausgenommen CP/M-Files und Textdateien)
- ermittelt Länge in KBytes von jedem Diskettenfile
- ermittelt Einsprungadresse (wenn vorhanden) von jedem Diskettenfile (ausgenommen CP/M-Files und Textdateien)
- ermittelt Filenamen überhaupt
- ermittelt Filetyp (&,',$,*,...)
- ermittelt die Anzahl von Files auf einer Diskette
- ermittelt das Format, mit dem die Diskette formatiert wurde.

Variablen von D-I-R:
Basis: Adresse im Directorybuffer, aus der die Filenamen und die Filelängen ausgelesen werden.
Status: Wenn das Programm nochmal durch den Benutzer durch ENTER gestartet wird, dann ist Status = -1 und das Programm weiß, daß es verschiedene Programmabschnitte nicht nochmal durchlaufen muß.
A$,B$: sind die Hexstrings, die aus den jeweiligen Adressen ausgelesen werden.
Name$: der Filename
nml: Länge des Filenamens
nmadr: Adresse, in der der Filename steht
flag: wenn flag=5 ist, dann steht im Directorybuffer fünfmal hintereinander eine Null, d.h., daß das Directory fertig ausgelesen ist.
f: Anzahl der gelesenen Files

Das eigentliche Geheimnis des Programms liegt darin: Wenn man in Assembler einen Openout durchführt, hat man schon in zwei Registern Start und Länge des Programms. Filetyp und Einsprungadresse werden einem nicht sofort auf dem Tablett serviert, man muß noch an den Registern herumbasteln, bis man auch diese hat. Das Programm hat einen kleinen Nachteil, es liest keine vom Assembler "GENA" erstellten Files richtig aus. Ansonsten gibt es keine Probleme. Selbst wenn das Programm gerade beim Auslesen von Files ist, kann man es beenden, indem man "E" bzw. "e" drückt.

M. Wegman

```
100 REM ********************************** [1887]
******************
110 REM * D.I.R.    Disc Information Reade [2563]
r               *
120 REM *             (C) 1986  by  WEGIwar [1050]
e               *
130 REM *           Written  by  M. WEGMAN [1653]
N               *
140 REM ********************************** [1887]
******************
150 IF STATUS=-1 THEN CLS:PRINT"           [7659]
   Diskette einlegen und RETURN/ENTER drue
cken !" :BASIS=&7000:FLAG=0:NAME$="":F=0 :
st=0:GOTO 460
160 INK 1,26:INK 0,26:PAPER 0:PEN 1:BORDER [2214]
 26:MODE 2
170 REM *****************************      [1471]
180 REM *   Windows   definieren    *      [1988]
190 REM *****************************      [1471]
200 WINDOW #6,80,80,5,22                   [1454]
210 WINDOW #5,1,80,22,22                   [1389]
220 WINDOW #4,1,2,5,22                     [1094]
230 WINDOW #3,1,80,4,4                     [1181]
240 WINDOW #2,1,80,23,25                   [1455]
250 WINDOW #1,1,80,1,4                     [1004]
260 WINDOW #0,2,79,6,19                    [1075]
270 basis=&7000:f=0:st=0                   [677]
280 REM ********************************** [1867]
*********
290 REM * Bildschirmmaske  /  Titelbild    [5642]
printen *
300 REM ********************************** [1867]
*********
310 :                                      [174]
320 PRINT #1,CHR$(150);:FOR t=1 TO 78:PRIN [3574]
T #1,CHR$(154);:NEXT:PRINT #1,CHR$(156);
330 PRINT #1,CHR$(149)"            D.I.R. [7618]
   Disc Information Reader   (C) 1986 by W
EGIware           "CHR$(149);
340 PRINT #1,CHR$(147);:FOR t=1 TO 78:PRIN [4607]
T #1,CHR$(154);:NEXT:PRINT #1,CHR$(153);
350 PRINT #2,CHR$(150);:FOR t=1 TO 78:PRIN [3723]
T #2,CHR$(154);:NEXT:PRINT#2,CHR$(156);
360 PRINT #2,CHR$(149)"           Anzahl [3131]
 der Files:"f"
```

```
                         "CHR$(149);
370 PRINT #2,CHR$(147);:FOR t=1 TO 78:PRIN [3476]
T #2,CHR$(154);:NEXT:PRINT #2,CHR$(153);
380 PRINT #3,CHR$(150);:FOR t=1 TO 78:PRIN [3019]
T #3,CHR$(154);:NEXT:PRINT#3,CHR$(156);
390 FOR t=1 TO 17:PRINT #4,CHR$(149):NEXT  [1924]
400 PRINT #5,CHR$(147);:FOR t=1 TO 78:PRIN [3799]
T #5,CHR$(154);:NEXT:PRINT #5,CHR$(153);
410 FOR t=1 TO 17:PRINT #6,CHR$(149);:NEXT [1716]
420 PRINT"              Diskette einlegen u [4483]
nd RETURN/ENTER druecken !"
430 GOSUB 920:'******* Maschinenkode einle [5257]
sen            *******
440 INK 1,0:REM ******* Bildschirmmaske si [4265]
chtbar machen *******
450 :                                       [174]
460 IF INKEY$ = CHR$(13) THEN 480          [1065]
470 GOTO 460                                [456]
480 LOCATE 1,1:FOR t=1 TO 78:PRINT" ";:NEX [2580]
T
490 :                                       [174]
500 REM *****************                  [1160]

510 REM * Hauptprogramm *                  [1191]
520 REM *****************                  [1160]
530 a=PEEK(&BB5A):POKE &BB5A,&C9:CALL &A03 [11096]
0:POKE &BB5A,a:REM ******* Directory lesen
 ohne dass es auf dem Bildschirm ausgedruc
kt wird *******
540 IF PEEK(&A89F)=65 THEN PRINT"Die Diske [4627]
tte hat CP/M Format":PRINT:GOTO 570
550 IF PEEK(&A89F)=193 THEN PRINT"Die Disk [4480]
ette hat DATA Format":PRINT:GOTO 570
560 PRINT"Die Diskette hat IBM Format":PRI [4075]
NT
570 IF PEEK(basis+11)=0 AND st=0 THEN 1280 [2951]
580 FOR t=1 TO 8:name$=name$+CHR$(PEEK(t+b [3074]
asis)):NEXT
590 f=f+1:REM ******* Fileanzahl erhoehen  [4688]
*******
600 name$=name$+"."                        [1463]
610 FOR t=9 TO 11:name$=name$+CHR$(PEEK(t+ [2835]
basis)):NEXT
620 PRINT name$;CHR$(149);                 [1052]
630 LOCATE #2,1,2:PRINT #2,CHR$(149)"      [4384]
          Anzahl der Files:"f"
640 adr=@name$                              [461]
650 nml=PEEK(adr)      :REM **** nml = Laen [2768]
ge des Namens ****
660 nmadr=PEEK(adr+1)+256*PEEK(adr+2)    :R [5142]
EM nmadr = Adresse , wo der Name liegt ***
*
670 CALL &A000,nmadr,nml                   [1718]
680 PRINT CHR$(PEEK(9015));CHR$(149);:IF P [6515]
EEK(9015)=42 THEN GOTO 1120 :REM **** Ausg
abe des Filetyps
690 pkadr=9000:GOSUB 1160      :REM **** ausl [5143]
esen der Speicheradressen 9000,9001 ****

700 GOSUB 1190     :REM **** -E- Taste gedru [3410]
eckt ? ****
710 PRINT"Startadresse &"b$;a$;CHR$(149);  [3430]
720 pkadr=9005:GOSUB 1160      :REM **** ausl [6835]
esen der Speicheradressen 9005,9006 ****
730 GOSUB 1190     :REM **** -E- Taste gedru [3410]
eckt ? ****
740 PRINT"Laenge &"b$;a$;CHR$(149);        [2867]
750 pkadr=9010:GOSUB 1160      :REM **** ausl [5250]
esen der Speicheradressen 9010,9011 ****
760 GOSUB 1190     :REM **** -E- Taste gedru [3410]
eckt ? ****
770 IF a$="00" AND b$="00" THEN a$="--":b$ [2195]
=a$
780 PRINT"Einsprungadr. &"b$;a$;CHR$(149); [2622]
790 PRINT"KByte:"PEEK(basis+12)            [3650]

800 GOSUB 1190                              [909]
810 :                                       [174]
820 REM ********************************** [1846]
*****
830 REM * Adresse vom naechsten Filenamen  [3884]
im    *
840 REM * Buffer suchen , auf Ende pruefen [3399]
      *
850 REM ********************************** [1846]
*****
860 basis=basis+11:flag=0                  [2163]
870 IF PEEK(basis)=255 THEN name$="":GOTO  [2475]
570
880 IF PEEK(basis)=0 THEN flag=flag+1:IF f [3583]
lag=5 THEN 1210:
890 REM **** Wenn 5 mal hintereinander ein [7449]
e Null im Buffer ist , dann Directory zu e
nde ->ENDE ****
900 basis=basis+1:GOSUB 1190               [1887]
910 GOTO 870                                [534]
911 :                                       [174]
912 REM **** Maschinensprache - Datas einl [4286]
esen ****
920 MEMORY &9FFF                            [134]
930 zaehler=0                              [1175]
940 READ a                                  [396]
950 IF a=-1 THEN RETURN                    [1108]
960 POKE &A000+zaehler,a                   [1184]
970 zaehler=zaehler+1                      [2165]
980 GOTO 940                                [312]
990 :                                       [174]
1000 DATA 254,002,192,221,102,003           [988]
1010 DATA 221,110,002,221,070,000          [1659]
1020 DATA 017,000,112,205,119,188          [1582]
1030 DATA 237,083,040,035,237,067          [1109]
1040 DATA 045,035,017,026,000              [1466]
1050 DATA 025,078,035,070,237,067          [1934]
1060 DATA 050,035,230,015,198,036          [1250]
1070 DATA 050,055,035,205,122,188          [1363]
1080 DATA 201                               [261]
1090 DATA &11,&00,&70,&cd,&9b,&bc          [1873]
1100 DATA &c9                               [347]
1110 DATA -01                               [284]
1120 PRINT"                   "CHR$(149)"  [5236]
          "CHR$(149)"                   "C
HR$(149)"KByte:"PEEK(basis+12 ):GOTO 860
1130 REM ********************************* [2675]
*******************************
1140 REM * Adressen auslesen und gegebenen [5226]
falls Hex - Strings auffuellen *
1150 REM ********************************* [2675]
*******************************
1160 st=1:a$=HEX$(PEEK(pkadr+0)):IF LEN(a$ [6158]
) <> 2 THEN c$="0":c$=c$+a$:a$=c$
1170 b$=HEX$(PEEK(pkadr+1)):IF LEN(b$) <>  [3991]
2 THEN c$="0":c$=c$+b$:b$=c$
1180 RETURN                                 [555]
1190 IF INKEY$="e" OR INKEY$="E" THEN 1210 [1308]
1200 RETURN                                 [555]
1210 CALL &BB03:REM **** Tastaturbuffer le [3400]
eren ****
1220 LOCATE #1,1,2:PRINT #1,CHR$(149)"     [5178]
             Press -RETURN- TO RESTART OR
 -E- TO END                        "CHR$(149)
;
1230 a$="":WHILE a$="":a$=INKEY$:WEND      [2835]
1240 IF a$="E" OR a$="e" THEN MODE 2:END   [1655]
1250 IF a$=CHR$(13) THEN 1270              [1787]
1260 GOTO 1230                              [365]
1270 STATUS=-1:LOCATE #1,1,2:PRINT #1,CHR$ [11577]
(149)"            D.I.R.   Disc Informati
on Reader    (C) 1986 by WEGIware
"CHR$(149); :GOTO 150
1280 PRINT"Kein File Vorhanden !":GOTO 121 [4020]
0
```

Disk-Manager

für 464-664-6128



Der Disk-Manager ist ein anwenderfreundliches Toolkit zur Bearbeitung von Directorys und Dateien.
Nach dem Starten mit RUN erscheint folgendes Menue:
Mit den Pfeiltasten ↑ ↓ läßt sich das ◀ vertikal positionieren. Wird gleichzeitig die CONTROL-Taste gedrückt, so springt das ◀ in die oberste bzw. in die unterste Zeile.
Beinhaltet die Directory mehr als 25 Eintragungen, so wird die Liste geteilt.

<R> .....RUN
startet ein mit ◀ bezeichnetes Programm

<L> .....LOAD
lädt ein mit ◀ bezeichnetes Programm

< C> .....CATDRUCK
gibt das Inhaltsverzeichnis auf dem Drucker aus (ist nur bei eingeschaltetem Drucker eingeblendet).
< C> .....DISKLIST
gibt das Inhaltsverzeichnis auf dem Drucker aus, wobei die Länge des Files (in K) angezeigt wird. (Ist nur bei eingeschaltetem Drucker eingeblendet.)
< U> .....USER
ermöglicht das Wechseln des USER-Bereichs (0 bis 15)
< F> .....FLOPPY
ermöglicht die Wahl des Laufwerks (A oder B)
< DEL> .....DELETE
verzweigt in das DELETE-MENUE, welches folgende Funktionen bereitstellt:
Das ◀ wird wie zuvor gesteuert.
< SPACE>
markiert ◀ als zu löschendes File mit * (wird durch zweimaligen Druck rückgängig gemacht).
< ENTER>
löscht alle markierten Files und geht zurück zum Hauptmenue
< E>
löscht alle markierten Files, führt den üblichen |ERA-Befehl aus und geht zurück zum Hauptmenue
CTRL < DEL>
löscht die gesamte Diskette im aktuellen USER-Bereich
CTRL < R> .....RENAME
dient zur Umbenennung von Files
< SPACE>
meldet eine neue Diskette an
< COPY>
kopiert den DISK-MANAGER
CTRL < C> .....CPM
< ENTER> .....BASIC
CTRL < D>
verzweigt in das Menue zur Druckereinstellung:
< N> NLQ ein/aus
< 1> Zeichensatz 1
< 2> Zeichensatz 2
< E> Engschrift ein/aus
< H> Zeichenhervorhebung ein/aus
< D> Doppelanschlag ein/aus
< V> Zeichenvergrößerung ein/aus
< M> Menue
CTRL < U>
Umlaute installieren
CTRL < F>
Funktionstasten belegen

Folgende Funktionen stehen nach Beendigung des Programms im Basic zur Verfügung:
f7.....NLQ ein
f8.....Normalschrift ein
f9.....List#8
f4.....LOAD "
f5.....RUN "
f6.....SAVE "
f1.....AUTO
f2.....MODE 2:LIST
f3.....CAT
f0.....CLS
TAB.....LINE FEED (Drucker)
< .> .....RUN
CTRL < T> .....TURBO DOS aktivieren

Jürgen Gergens

```
10 '     D I S C - M A N A G E R      VERSI [2523]
ON 6.0/464-6128
20 '     (c) 1986 by Juergen Gerkens        [2627]
30 '     ---------------------------        [1412]
40 CALL &BB00:POKE &AC00,1:SYMBOL AFTER 25 [5445]
6:KEY 12,"RUN"+CHR$(34)+"#"+CHR$(34)+CHR$(
13)
50 |USER,0:|DISC:ON ERROR GOTO 100          [1820]
60 IN$=CHR$(24)                             [636]
70 FOR I=47840 TO 47848:READ A:POKE I,A:NE [3359]
XT I
80 DATA 205,96,187,50,240,186,201,0,0       [1366]
90 DIM F$(65),S(65)                         [767]
100 PEN 1:PAPER 0:PEN#1,0:PAPER#1,1:MODE 2 [2583]
:CAT:N=0:T=0
110 LOCATE 7,2:CALL &BAE0:fl$=CHR$(PEEK(&B [3302]
AF0))
120 LOCATE 15,2:CALL &BAE0:us$=CHR$(PEEK(& [5783]
BAF0)):PRINT CHR$(9);:CALL &BAE0:us$=us$+C
HR$(PEEK(&BAF0)):us=VAL(us$)
130 FOR Z=1 TO 61 STEP 20:FOR Y=4 TO 19:N= [3302]
N+1:T=MAX(T,Y):F$(N)="":S(N)=0
140 FOR X=Z TO Z+16                         [1003]
150 LOCATE X,Y:CALL &BAE0:F$(N)=F$(N)+CHR$ [3045]
(PEEK(&BAF0)):IF F$(N)=" "THEN 170
160 NEXT X:NEXT Y                           [980]
170 F$(N)="":N=N-1:NEXT Z:T=T+1             [1307]
180 LOCATE 1,T:A$="":FOR Z=1 TO 10:CALL &B [4473]
AE0:A$=A$+CHR$(PEEK(&BAF0)):PRINT CHR$(9);
:NEXT Z
190 IF INSTR(1,F$(1),".")=0 AND F$(1)<>""T [1743]
HEN 100
200 WINDOW#0,1,50,1,25:WINDOW#1,51,80,1,25 [1745]
210 CLS#1:CLS                               [554]
220 PRINT#1,"   D I S K - M A N A G E R":P [4121]
RINT#1,"-----------------------------";
230 PRINT#1,"<R>......RUN"                  [1702]
240 PRINT#1,"<L>......LOAD"                 [2458]
250 IF INP(&F500)=90 THEN 270 ELSE PRINT#1 [2587]
,"<C>......CATDRUCK"
260 PRINT#1,"<D>......DISKLIST"             [1588]
270 PRINT#1,"<U>......USER"                 [1474]
280 PRINT#1,"<F>......FLOPPY":PRINT#1       [2384]
290 PRINT#1,"<DEL>....DELETE"               [1767]
300 PRINT#1,"<";IN$;"R";IN$;">......RENAME [1975]
"
310 PRINT#1,"<SPACE>..neue Disk anmelden"  [2848]
320 PRINT#1,"<COPY>...DISKMANAGER kopieren [3071]
":PRINT#1
330 PRINT#1,"<";IN$;"C";IN$;">......CPM"   [2446]
340 PRINT#1,"<ENTER>..BASIC":PRINT#1       [2883]
350 IF INP(&F500)=90 THEN 360 ELSE PRINT#1 [6242]
,"<";IN$;"D";IN$;">......Drucker einstelle
n"
360 PRINT#1,"<";IN$;"U";IN$;">......Umlaut [3918]
e installieren"
370 PRINT#1,"<";IN$;"F";IN$;">......Funkti [3037]
onstasten"
380 PRINT#1,"<";IN$;"T";IN$;">......TURBO  [2877]
DOS":PRINT#1
390 LOCATE#1,23,3:PRINT#1,CHR$(240);" ";CH [6482]
R$(241);CHR$(8);CHR$(8);CHR$(8);CHR$(10);C
HR$(10);IN$;CHR$(240);IN$;" ";IN$;CHR$(241
);IN$
400 LOCATE#1,2,23:PRINT#1,in$;:PRINT#1,USI [8197]
NG"Drive \\: user ##: \          \";fl$,us,a$
;:PRINT#1,in$
410 LOCATE#1,1,25:PRINT#1,"  ";CHR$(164);" [3229]
 1986 by Juergen Gerkens"
420 A=1:B=25:Z=1                            [1257]
430 GOSUB 1760:CLS:FOR W=A TO MIN(B,N):PRI [3611]
NT F$(W):NEXT W
440 LOCATE 25,Z-A+1:PRINT CHR$(247)         [2009]
450 CALL &BB06                              [393]
460 LOCATE 25,Z-A+1:PRINT" "                [1734]
470 IF INKEY(2)=0 THEN Z=MIN(Z+1,N):IF Z>B [4512]
 THEN A=A+25:B=B+25:Z=A:GOTO 430
480 IF INKEY(0)=0 THEN Z=MAX(Z-1,1):IF Z<A [3031]
 THEN A=A-25:B=B-25:Z=B:GOTO 430
490 IF INKEY(2)>0 THEN Z=MIN(Z+25,N):IF Z> [5185]
B THEN A=A+25:B=B+25:Z=A:GOTO 430
500 IF INKEY(0)>0 THEN Z=MAX(Z-25,1):IF Z< [4458]
A THEN A=A-25:B=B-25:Z=B:GOTO 430
510 IF INKEY(50)=0 THEN MODE 2:RUN LEFT$(F [1898]
$(Z),12)
520 IF INKEY(36)=0 THEN MODE 2:LOAD LEFT$( [2982]
F$(Z),12)
530 IF INKEY(62)=0 THEN GOTO 1440           [1354]
540 IF INKEY(61)=0 THEN GOTO 1570           [1761]
550 IF INKEY(42)=0 THEN CLS#1:INPUT#1,"USE [4405]
R-NR.:",T:T=MIN(15,INT(ABS(T))):|USER,T:GO
TO 100
560 IF INKEY(53)=0 THEN CLS#1:PRINT#1,"FLO [3183]
PPY :"ELSE 600
570 IF INKEY(69)=0 THEN|A:GOTO 100          [1770]
580 IF INKEY(54)=0 THEN|B:GOTO 100          [840]
590 GOTO 570                                [379]
600 IF INKEY(79)=0 THEN GOTO 1210           [1364]
610 IF INKEY(50)>127 THEN LOCATE 1,Z-A+1:I [3597]
NPUT"",B$:C$=LEFT$(F$(Z),12):|REN,@B$,@C$:
GOTO 100
620 IF INKEY(47)=0 THEN|USER,0:GOTO 100    [1749]
630 IF INKEY(62)>127 THEN MODE 2:|CPM      [2075]
```

```
640 IF INKEY(9)=0 THEN SAVE"#":GOTO 100           [329]
650 IF INKEY(61)>127 THEN GOTO 950                [1245]
660 IF INKEY(42)>127 THEN GOSUB 850               [1308]
670 IF INKEY(53)>127 THEN GOSUB 710               [1660]
680 IF INKEY(51)>127 THEN GOSUB 1700              [1583]
690 IF INKEY(6)=0 OR INKEY(18)=0 THEN MODE        [1385]
 2:NEW
700 GOTO 440                                      [444]
710 KEY 7,"?#8,CHR$(27);"+CHR$(34)+"x1"+CH        [4344]
R$(34)+":'NLQ"+CHR$(13)
720 KEY 8,"?#8,CHR$(27);"+CHR$(34)+"x2"+CH        [2432]
R$(34)+":' NORM"+CHR$(13)
730 KEY 9,"list#8"                                [950]
740 KEY 4,"load"+CHR$(34)                         [611]
750 KEY 5,"run"+CHR$(34)                          [714]
760 KEY 6,"save"+CHR$(34)                         [1073]
770 KEY 1,"auto "                                 [769]
780 KEY 2,"mode 2:list "                          [994]
790 KEY 3,"CAT"                                   [477]
800 KEY 0,"CLS"+CHR$(13)                          [1698]
810 KEY 10,"RUN"+CHR$(13)                         [1945]
820 KEY 141,"?#8"+CHR$(13):KEY DEF 68,1,14        [2116]
1
830 KEY 12,"|DISC:RUN"+CHR$(34)+"#"+CHR$(3        [2682]
4)+CHR$(13)
840 RETURN                                        [555]
850 SYMBOL AFTER 90                               [1214]
860 SYMBOL 91,&5A,&3C,&66,&66,&7E,&66,&66,        [2260]
&0
870 SYMBOL 92,&BA,&6C,&C6,&C6,&C6,&6C,&38,        [2305]
&0
880 SYMBOL 93,&66,&0,&66,&66,&66,&66,&3C,&        [2316]
0
890 SYMBOL 123,&48,&0,&78,&C,&7C,&CC,&76,&        [1768]
0
900 SYMBOL 124,&24,&0,&3C,&66,&66,&66,&3C,        [2269]
&0
910 SYMBOL 125,&44,&0,&66,&66,&66,&66,&3E,        [1841]
&0
920 SYMBOL 126,&38,&6C,&6C,&6C,&66,&76,&6C        [2642]
,&60
930 KEY DEF 26,1,125,93,124:KEY DEF 17,1,1        [3363]
24,92:KEY DEF 19,1,123,91
940 RETURN                                        [555]
950 IF INP(&F500)=90 THEN 200 ELSE MODE 2:        [3730]
WINDOW#0,25,60,5,25
960 PRINT"Drucker einstellen:":PRINT"-----        [5547]
---------------":PRINT:PRINT
970 PRINT"<N>...NLQ                  ein":PRIN    [2700]
T
980 PRINT"<1>...Zeichensatz 1          aus"       [2742]
990 PRINT"<2>...Zeichensatz 2        ein":PRIN    [2919]
T
1000 PRINT"<E>...Engschrift              aus"     [2432]
1010 PRINT"<H>...Hervorhebung            aus"     [3060]
1020 PRINT"<D>...Doppelanschlag          aus"     [3056]
1030 PRINT"<V>...Vergroesserung          aus"     [2647]
:PRINT
1040 PRINT"<M>...Menue"                           [834]
1050 GOSUB 1760:DN=1:DE=0:DH=0:DD=0:DV=0          [1360]
1060 CALL &BB06:IF INP(&F500)=90 THEN 1190        [1616]
1070 IF INKEY(46)<>0 THEN 1090 ELSE DN=1-D        [1795]
N
1080 IF DN=1 THEN LOCATE 24,5:PRINT"ein           [6462]
 ":PRINT#8,CHR$(27);"x1"ELSE LOCATE 24,5:P
RINT"     aus":PRINT#8,CHR$(27);"x2"
1090 IF INKEY(64)=0 THEN LOCATE 24,7:PRINT        [3996]
"ein     ":LOCATE 24,8:PRINT"     aus":PRINT
#8,CHR$(27);CHR$(55)
1100 IF INKEY(65)=0 THEN LOCATE 24,7:PRINT        [5256]
"     aus":LOCATE 24,8:PRINT"ein      ":PRINT
#8,CHR$(27);CHR$(54)
1110 IF INKEY(58)<>0 THEN 1130 ELSE DE=1-D        [712]
E
1120 IF DE=1 THEN LOCATE 24,10:PRINT"ein          [5469]
  ":PRINT#8,CHR$(15)ELSE LOCATE 24,10:PRIN
T"     aus":PRINT#8,CHR$(18)
1130 IF INKEY(44)<>0 THEN 1150 ELSE DH=1-D        [1136]
H
1140 IF DH=1 THEN LOCATE 24,11:PRINT"ein          [6532]
  ":PRINT#8,CHR$(27);CHR$(69)ELSE LOCATE 2
4,11:PRINT"      aus":PRINT#8,CHR$(27);CHR$(
70)
1150 IF INKEY(61)<>0 THEN 1170 ELSE DD=1-D        [1593]
D
1160 IF DD=1 THEN LOCATE 24,12:PRINT"ein          [6901]
  ":PRINT#8,CHR$(27);CHR$(71)ELSE LOCATE 2
4,12:PRINT"      aus":PRINT#8,CHR$(27);CHR$(
72)
1170 IF INKEY(55)<>0 THEN 1190 ELSE DV=1-D        [2264]
V
1180 IF DV=1 THEN LOCATE 24,13:PRINT"ein          [7445]
  ":PRINT#8,CHR$(27);CHR$(87);"1"ELSE LOCA
TE 24,13:PRINT"     aus":PRINT#8,CHR$(27);C
HR$(87);"0"
1190 IF INKEY(38)=0 OR INKEY(18)=0 OR INKE        [3701]
Y(6)=0 OR INKEY(47)=0 THEN 200
1200 GOTO 1060                                    [311]
1210 CLS#1                                        [373]
1220 PRINT#1,"DELETE:":PRINT#1,"-------":P        [2911]
RINT#1
1230 PRINT#1,"<SPACE>..markieren":PRINT#1         [2983]
1240 PRINT#1,CHR$(240);"  ";CHR$(241);"  "        [4745]
;IN$;CHR$(240);IN$;"  ";IN$;CHR$(241);IN$:
PRINT#1
1250 PRINT#1,"<ENTER>..DELETE":PRINT#1            [1828]
1260 PRINT#1,"<E>.....ERA":PRINT#1                [2218]
1270 PRINT#1,"<";IN$;"DEL";IN$;">....Diske        [4813]
tte loeschen":PRINT#1
1280 GOTO 1300                                    [438]
1290 GOSUB 1760:CLS:FOR W=A TO MIN(B,N):PR        [4993]
INT F$(W);"    ";CHR$(32+s(w)*10):NEXT W
1300 LOCATE 25,Z-A+1:PRINT CHR$(247)              [2009]
1310 CALL &BB06                                   [393]
1320 LOCATE 25,Z-A+1:PRINT" "                     [1734]
1330 IF INKEY(2)=0 THEN Z=MIN(Z+1,N):IF Z>        [4476]
B THEN A=A+25:B=B+25:Z=A:GOTO 1290
1340 IF INKEY(0)=0 THEN Z=MAX(Z-1,1):IF Z<        [3025]
A THEN A=A-25:B=B-25:Z=B:GOTO 1290
1350 IF INKEY(2)>0 THEN Z=MIN(Z+25,N):IF Z        [5309]
>B THEN A=A+25:B=B+25:Z=A:GOTO 1290
1360 IF INKEY(0)>0 THEN Z=MAX(Z-25,1):IF Z        [4550]
<A THEN A=A-25:B=B-25:Z=B:GOTO 1290
1370 IF INKEY(47)=0 THEN S(Z)=1-S(Z):LOCAT        [4658]
E 23,Z-A+1:IF S(Z)=1 THEN PRINT"*"ELSE PRI
NT" "
1380 IF INKEY(79)>0 THEN C$="*.*":|ERA,@C$        [2016]
:GOTO 100
1390 IF INKEY(18)=0 OR INKEY(6)=0 THEN GOS        [1886]
UB 1420:GOTO 100
1400 IF INKEY(58)=0 THEN CLS#1:INPUT#1,"ER        [5180]
A:",A$:GOSUB 1420:|ERA,@A$:GOTO 100
1410 GOTO 1300                                    [438]
1420 FOR Z=1 TO N:IF S(Z)=1 THEN C$=LEFT$(        [3243]
F$(Z),12):|ERA,@C$
1430 NEXT Z:RETURN                                [788]
1440 IF INP(&F500)=90 THEN 200                    [1157]
1450 CLS#1:INPUT#1,"NR.    :",T                   [1253]
1460 INPUT#1,"TITEL.:",T$                         [946]
1470 FOR Z=N+1 TO 64:F$(Z)="":NEXT Z              [1868]
1480 PRINT#8,CHR$(15);CHR$(27);"x2"               [2061]
1490 PRINT#8,STR$(T);") ";T$                      [1865]
1500 PRINT#8,STRING$(60,"-")                      [1772]
1510 IF us THEN PRINT#8,USING"user ##";us         [1986]
1520 FOR Z=1 TO N STEP 4                          [1472]
1530 PRINT#8,USING"\              \     \         [4687]
   \      \            \      \           \";F$(
Z),F$(Z+1),F$(Z+2),F$(Z+3)
1540 NEXT Z:PRINT#8:PRINT#8,A$:PRINT#8:PRI        [4890]
NT#8:PRINT#8:PRINT#8:PRINT#8
1550 PRINT#8,CHR$(18)                             [1094]
1560 GOTO 200                                     [429]
1570 IF INP(&F500)=90 THEN 200                    [1157]
1580 CLS#1:INPUT#1,"NR.    :",T                   [1253]
1590 INPUT#1,"TITEL.:",T$                         [946]
1600 FOR Z=N+1 TO 64:F$(Z)="":NEXT Z              [1868]
1610 PRINT#8,CHR$(15);CHR$(27);"x2"               [2061]
1620 PRINT#8,STR$(T);") ";T$                      [1865]
1630 PRINT#8,STRING$(83,"-")                      [1521]
1640 IF us THEN PRINT#8,USING"user ##";us         [1986]
1650 FOR Z=1 TO N STEP 4                          [1472]
1660 PRINT#8,USING"\                   \       \  [5722]
             \       \                  \
 \              \";F$(Z),F$(Z+1),F$(Z+2),
F$(Z+3)
1670 NEXT Z:PRINT#8:PRINT#8,A$:PRINT#8:PRI        [4890]
NT#8:PRINT#8:PRINT#8:PRINT#8
1680 PRINT#8,CHR$(18)                             [1094]
1690 GOTO 200                                     [429]
1700 FOR P=&A000 TO &A012                         [1122]
1710 READ T$:POKE P,VAL("&"+T$)                   [872]
1720 NEXT P                                       [364]
1730 CALL &A000                                   [637]
1740 RETURN                                       [555]
1750 DATA 21,A,A0,DF,7,A0,C9,D,C6,7,23,0,C        [2275]
8,0,1,1,A,0,3,0
1760 C$=INKEY$:WHILE C$<>"":C$=INKEY$:WEND        [2666]
:RETURN
```

AMSDOS INTERN

Fortsetzung von S. 23

Abbildung 2

```
A000 00 54 45 53 54 20 20 20 20 42 41 53 00 00 00  .TEST    BAS...
A00F 02 02 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
A01E 00 00 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ..eeeeeeeeeeeee
A02D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A03C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A04B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A05A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A069 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A078 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A087 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A096 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0A5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0B4 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0C3 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0D2 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0E1 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0F0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A0FF E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A10E E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A11D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A12C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A13B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A14A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A159 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A168 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A177 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A186 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A195 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1A4 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1B3 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1C2 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1D1 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1E0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1EF E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A1FE E5 E5 00 00 00 00 00 00 00 00 00 00 00 00 00  ee.............
```

Abbildung 3

```
           1      2                       3       4
A000 00 54 45 53 54 20 20 20 20 42 41 53 00 00 00  .TEST    BAS...
A00F 02 02 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
A01E 00 00 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ..eeeeeeeeeeeee
A02D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A03C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A04B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
A05A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeeeeeeeeee
5
6
```

Danach sieht der Directorysektor wie in Abbildung 2 aus.

In den beiden ersten 16 Zeichenreihen, findet Amsdos alle wichtigen Directoryinformationen über die Datei TEST.BAS. Was diese 32 Bytes genau bedeuten, wollen wir nun klären.

Dabei läßt sich der Directoryeintrag in sechs Teile gliedern. Sehen Sie sich den Eintrag einmal in Abbildung 3 an.

Die Zahlen der einzelnen Sektionen haben folgende Bedeutung:

1. Das erste Byte eines jeden Directoryeintrags legt fest, auf welcher Benutzerebene sich das Programm befindet. Ist der Wert dieses Bytes 00, so befindet sich das File auf der USER-Nummer 0. Eine Sonderregelung findet hier bei dem Wert &E5 statt. Hat das erste Byte diesen Wert, so wurde es mit ERA gelöscht.

2. Die nächsten acht Bytes &01-&08 enthalten den Dateinamen, der maximal acht Zeichen lang sein kann.

3. In den Bytes &09-&0B befindet sich die Typenkennzeichnung des Files [BIN,BAS,BAK...].

Außerdem wird durch ein gesetztes Bit 7 in den Bytes &09, &0A das Fileattribut angegeben. Dabei gilt Folgendes:

Abbildung 4

Byte &09:	BIT 7=0	PROGRAMM KANN GELÖSCHT WERDEN
	BIT 7=1	PROGRAMM IST READ ONLY UND KANN NICHT GELÖSCHT WERDEN
BYTE &0A	BIT 7=0	PROGRAMM IST IM DIRECTORY SICHTBAR
	BIT 7=1	PROGRAMM IST EIN SYS FILE UND KANN NICHT IM DIRECTORY GELISTET WERDEN

4. Das Byte &0C gibt eine Referenz zu den Bytes &10-&1F und kennzeichnet den Extend.

5. In dem Byte &0F wird die Länge des Programms angegeben. Dabei entspricht der Wert von &0F der Anzahl 128 Bytes Records, die das Programm belegt.

6. Die letzten 16 Bytes (&10-&1F) sind für das Amsdos mit am interessantesten. Hier kann man nachlesen, auf welchem Track und Sektor die einzelnen Blöcke zu finden sind. Die Blöcke werden, wie anfangs erwähnt, durchnumeriert. Die 16 Ein-Byte-Zahlen geben also die Reihenfolge und Lage der 1 KB großen Programmteile an. Den genauen Standort eines Blocks auf der Diskette ermittelt man durch die beiden folgenden Formeln:

TRACK = INT ((BLOCKNUMMER*2+19)/9)
SEKTOR = (BLOCKNUMMER*2+18) MOD 9

Da so ein Directoryeintrag nur 16 Blocknummern halten kann, stellt sich natürlich die Frage, was passiert, wenn ein Programm größer ist als 16 * 1 KB. Das Amsdos bedient sich dabei einer recht genialen Lösung. Es setzt einfach den Wert des Bytes &0C (EXTEND) auf den Wert &80 und legt einen zweiten Directoryeintrag an. Findet der Controller beim Laden in der Directory das Byte &0C mit dem Wert &80 beschrieben, so sucht er sich, nachdem er 16 Blöcke geladen hat, den nächsten gleichnamigen Directoryeintrag und lädt so lange weiter, bis er im Blockverzeichnis (BAM-BLOCK ALLOCTATION MAP) den Wert 00 findet und hiermit das Ende der Datei erreicht ist.

Damit wäre dann auch schon alles über die Directoryeinträge auf der Diskette gesagt. Woher das Programm nun weiß, wohin im Speicher ein Programm geladen werden soll, erklärt sich bei der Analyse des ersten Blocks der Datei, der die Header-Info enthält.

Der Dateikopf

Die erste Blocknummer unserer Testdatei hatte den Wert 02. Wir können nun mit unseren beiden Formeln die Position (TRACK 2/SEKTOR 4) errechnen. Nun müssen wir nur noch auf den, für System-Format typischen, Offset von &41 zum Sektor addieren und geben ein:

IREAD,0,2,&45
IEXAMINE,0

Das Bild 5 zeigt einen Ausdruck des Sektors, auf dem sich der Programmkopf von TEST.BAS sowie das eigentliche Programm befindet.

Abbildung 5

```
A000 00 54 45 53 54 20 20 20 20 42 41 53 00 00 00 .TEST    BAS...
A00F 00 00 00 00 00 00 70 01 00 61 00 00 00 00 00 ......p..a.....
A01E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
A02D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
A03C 00 00 00 00 61 00 00 C9 03 18 00 0A 00 C5 20 ....a..I.....E
A04B 44 49 45 53 20 49 53 54 20 45 49 4E 20 54 45 DIES IST EIN TE
A05A 53 54 00 20 00 14 00 C5 20 55 4D 20 5A 55 20 ST. ...E UM ZU
A069 53 45 48 45 4E 20 57 49 45 20 44 49 45 20 44 SEHEN WIE DIE D
A078 41 54 45 4E 00 27 00 1E 18 00 0A 00 C5 20 44 ATEN.'......E D
A087 49 45 53 20 49 53 54 20 45 49 4E 20 54 45 53 IES IST EIN TES
A096 54 00 20 00 14 00 C5 20 55 4D 20 5A 55 20 53 T. ...E UM ZU S
A0A5 45 48 45 4E 20 57 49 45 20 44 49 45 20 44 41 EHEN WIE DIE DA
A0B4 54 45 4E 00 27 00 1E 00 C5 20 41 55 46 20 44 TEN.'...E AUF D
A0C3 45 52 20 44 49 53 4B 45 54 54 45 20 41 42 47 ER DISKETTE ABG
A0D2 45 4C 45 47 54 20 57 45 52 44 45 4E 00 00 00 ELEGT WERDEN...
A0E1 1A 43 52 20 20 20 20 0D 0A 44 45 46 4D 20 22 .CR    ..DEFM "
A0F0 46 4F 52 4D 41 54 54 49 4E 47 2E 2E 2E 50 4C FORMATTING...PL
A0FF 45 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 Eeeeeeeeeeeeee
A10E E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A11D E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A12C E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A13B E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A14A E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A159 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A168 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A177 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A186 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A195 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1A4 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1B3 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1C2 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1D1 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1E0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1EF E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
A1FE E5 E5 00 00 00 00 00 00 00 00 00 00 00 00 00 ee.............
```

Auch hier kann man wieder eine Unterteilung in verschiedene Sektionen vornehmen.

Abbildung 6

```
A000 00 54 45 53 54 20 20 20 20 42 41 53 00 00 00 .TEST    BAS...
A00F 00 00 00 00 00 00 70 01 00 61 00 00 00 00 00 ......p..a.....
A01E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
A02D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
A03C 00 00 00 00 61 00 00 C9 03 18 00 0A 00 C5 20 ....a..I.....E
A04B 44 49 45 53 20 49 53 54 20 45 49 4E 20 54 45 DIES IST EIN TE
A05A 53 54 00 20 00 14 00 C5 20 55 4D 20 5A 55 20 ST. ...E UM ZU
A069 53 45 48 45 4E 20 57 49 45 20 44 49 45 20 44 SEHEN WIE DIE D
A078 41 54 45 4E 00 27 00 1E 18 00 0A 00 C5 20 44 ATEN.'......E D
A087 49 45 53 20 49 53 54 20 45 49 4E 20 54 45 53 IES IST EIN TES
A096 54 00 20 00 14 00 C5 20 55 4D 20 5A 55 20 53 T. ...E UM ZU S
A0A5 45 48 45 4E 20 57 49 45 20 44 49 45 20 44 41 EHEN WIE DIE DA
A0B4 54 45 4E 00 27 00 1E 00 C5 20 41 55 46 20 44 TEN.'...E AUF D
A0C3 45 52 20 44 49 53 4B 45 54 54 45 20 41 42 47 ER DISKETTE ABG
A0D2 45 4C 45 47 54 20 57 45 52 44 45 4E 00 00 00 ELEGT WERDEN...
A0E1 1A 43 52 20 20 20 20 0D 0A 44 45 46 4D 20 22 .CR    ..DEFM "
A0F0 46 4F 52 4D 41 54 54 49 4E 47 2E 2E 2E 50 4C FORMATTING...PL
A0FF 45 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 Eeeeeeeeeeeeee
A10E E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 eeeeeeeeeeeeeee
```


Zunächst fällt einmal auf, daß in den Bytes &01-&0B wieder der Programmname und die Kennung stehen. Da dieser Name kaum etwas mit den abgespeicherten Daten zu tun hat, kann man davon ausgehen, daß sich hier in den ersten 128 Bytes der Record für den Programmkopf befindet. Tatsächlich wird das unter Amsdos auch so gehalten. Das eigentliche Programm steht erst ab Byte &80 und dürfte für unsere Analyse uninteressant sein.
Wesentlich wichtiger ist der Programmkopf, in dem natürlich noch viel mehr Informationen als nur der Name des Programms stehen.
Folgende Bytes machen dabei wichtige Aussagen über das Programm:

BYTE &00
USER NUMMER

BYTE &01-&08
PROGRAMMNAME
BYTE &09-&0B
KENNZEICHNUNG

BYTE &13
ABGESPEICHERT ALS
&00 NORMALES SAVE
&01 GESCHÜTZT GESAVED
&02 BINÄR GESAVED

BYTE &15-&16
LADEADRESSE DES PROGRAMMS (IN UNSEREM FALL &0170- BASIC-ANFANG)

BYTE &18-&19
LÄNGE DES PROGRAMMS

BYTE &1B-&1C
EXECUTION ADRESSE BEI MC-PROGRAMMEN

BYTE &40-&41
LÄNGE DES PROGRAMMS (WARUM EIGENTLICH ZWEIMAL ?)

BYTE &43-&44
CHECKSUMME

Bis auf die zweite Angabe der Programmlänge, deren Grund uns auch unbekannt ist, sowie die Checksumme, sollte es zu den Angaben eigentlich keine Fragen geben.

Die Checksumme addiert die Bytes &00-&42 und überprüft sie mit dem Eintrag in &43-&44. Anhand dieser Summe kann das Programm feststellen, ob eine Programmdatei oder ein ASCII-File vorliegt. ASCII-Files haben nämlich keinen Programmkopf und ergeben in diesem Fall einen Fehler.

Die Daten in Record 2 des Sektors werden in ASCII-Darstellung abgelegt und sind für den Anwender, soweit keine Modifikationen am Programm vorgenommen werden sollen, uninteressant. Wird das Programm jedoch mit SAVE"NAME",P abgespeichert, so werden nur die entsprechenden Token abgelegt, was ziemlich unübersichtlich wirkt. Ob der Ladevorgang abgeschlossen werden kann bzw. das Ende der Datei erreicht ist, erkennt der Rechner an einer Folge von 3-Null-Bytes. Alles, was sich weiterhin in den Records befindet, kann als Garbage bezeichnet werden und hat überhaupt keinen Einfluß auf das Programm.

Zu den formattypischen Offsets, die zur Sektornummer addiert werden müssen, sei noch gesagt, daß diese bei den einzelnen Formaten variieren und man die nachfolgende Tabelle als Grundlage zum Arbeiten benutzen kann. Bei exotischen Offsets hilft nur Probieren oder ein Sektor-Scanner.

FORMAT	STANDARD-OFFSET
SYSTEM	&41
VENDOR	&41
DATA ONLY	&C1
IBM	&01

Ausgerüstet mit diesen wichtigen Informationen über das Disk-Handling des Amsdos, sollten Sie eigentlich keine Schwierigkeiten mehr haben, sich auf Ihren Disketten zurechtzufinden, Modifikationen an Programmen vorzunehmen oder Disketten gegen unerwünschtes Kopieren zu schützen.

Probieren Sie das Ganze einmal mit Binär- oder ASCII-Dateien, sowie mit einem anderen Format. Sobald Sie ein wenig Übung im Umgang mit der doch recht simplen Diskettenorganisation haben, wird Ihnen Ihre DDI-1 gleich in einem ganz anderen Licht erscheinen.
(TM)

Ein universelles Dateiverwaltungs-programm für den CPC 6128

für 6128



Mit UNIDAT können Sie jede denkbare Ansammlung von Daten komfortabel verwalten; von der Adressenverwaltung bis zur Zeitschriftensammlung. Der Menuepunkt 'Neue Datei erstellen' bietet die Möglichkeit, sich einen elektronischen Karteikasten maßzuschneidern, der genau den Anforderungen der gewünschten Datei entspricht. So können pro Datensatz (=Karteikarte) bis zu 15 Felder (=Überschriften wie 'Name', 'Adresse', 'PLZ' usw.) definiert werden, von welchen jedes wiederum 20 Zeichen lang sein darf.

Damit das Programm abgedruckt werden konnte, mußten wir es relativ kurz halten. So wurde auf einige Sicherheitsabfragen verzichtet, damit der Komfort erhalten blieb. Es ist daher empfehlenswert, den Hinweisen während des Programmablaufs zu folgen. Es muß auch darauf geachtet werden, daß der Dateiname zum Abspeichern einer Datei nicht länger als acht Zeichen ist, da sonst das Diskettenlaufwerk eine Fehlermeldung produziert.

Bevor UNIDAT geladen werden kann, muß unbedingt der auf der Systemdiskette des 6128 vorhandene 'Bankman' geladen und gestartet werden, weil die eingegebenen Daten während der Arbeit mit UNIDAT in der freien Speicherbank des 6128 angelegt werden. UNIDAT fragt zur Sicherheit am Programmanfang noch einmal ab, ob der Bankmanager geladen wurde.

Das Hauptmenue, das den Bediener durch sämtliche Funktionen führt, enthält zehn Auswahlmöglichkeiten, die nachstehend aufgeführt und erklärt sind:

1. Daten eingeben:
Hier wird gefragt, ob eine bestehende oder eine neue Datei bearbeitet werden soll. Eine bestehende Datei wird nach Angabe des Namens geladen, eine neue Datei wird nach den Wünschen des Bedieners erstellt. Anschließend können Daten neu eingegeben, verändert oder gelöscht werden. Nach der Arbeit kehrt man ins Hauptmenue zurück. Empfehlenswert ist, anschließend die Sortierroutine aufzurufen, welche die neuen Daten in die Datei einordnet.

2. Daten lesen:
Dieser Menuepunkt erlaubt ein sehr komfortables Blättern im 'Karteikasten'. Es besteht die Möglichkeit, die Daten der Reihe nach anzusehen oder bestimmte Daten suchen zu lassen, wobei auch die Eingabe eines Jokers ('?') möglich ist; wird z.B. im Feld 'Name' nach 'M?' gesucht, so findet UNIDAT 'Meier', 'Meyer', 'Maier' usw. Anschließend zurück ins Hauptmenue.

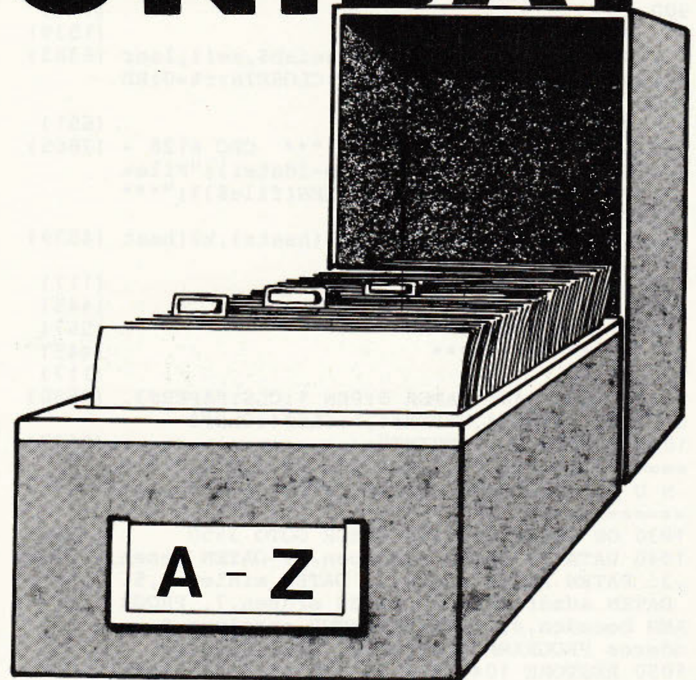
3. Daten einlesen,
4. Daten speichern:
Dieses sind die beiden Diskoperationen im Zusammenhang mit der Dateiarbeit. 'Daten einlesen' ermöglicht das Einladen einer auf Disk bestehenden Datei, wobei noch abgefragt wird, ob die gerade bestehende Datei vorher gespeichert werden soll. Bei 'Daten speichern' besteht die Möglichkeit, die eingelegte Diskette auf ausreichenden Speicherplatz zu überprüfen, und eine auf Disk gespeicherte Datei zu ändern oder zu löschen.

5. Daten ausdrucken:
Die Druckroutine arbeitet mit Epson-kompatiblen Druckern und erlaubt zu Beginn die Auswahl einiger Schriftarten und die Formatierung des Ausdrucks. Man kann die ganze Datei oder eine anwählbare Teildatei als Listing oder Adressaufkleber ausdrucken lassen. Die Aufkleber lassen sich ein- oder zweispaltig bedrucken.

6. Daten ordnen:
Dieser Menuepunkt dient dazu, eine Datei oder neu eingegebene Daten alphanumerisch zu sortieren. Hierzu wird das zu ordnende Feld abgefragt (Name, Adresse etc.). Anhand

```
10 '************************************** [1760]
******
20 '************************************** [1760]
******
30 '***                                     [51]
 ***
40 '***        C P C 6128 -  D A T E I      [1008]
 ***
50 '***     (c) by Guenter Treusch  DEZ.1985 [1653]
 ***
80 '***                                     [51]
 ***
90 '************************************** [1760]
******
100 '************************************** [1760]
*******
110 '                                       [117]
120 '                                       [117]
130 '                                       [117]
140 CLEAR:MODE 1:BORDER 1:INK 0,1:INK 1,26 [4691]
,1:INK 2,24:INK 3,6,1:PAPER 0:PEN 2:SPEED
INK 30,10
150 PRINT CHR$(7);"Haben Sie das Programm  [10280]
":PAPER 3:PEN 1:LOCATE 4,4:PRINT CHR$(7);"
 B A N K M A N ";:PAPER 0:PEN 2:PRINT "
eingelesen ?"
160 PRINT:PRINT:PRINT CHR$(24);CHR$(7);" W [5480]
enn ja, beliebige Taste druecken. ";CHR$(2
4)
170 IF INKEY$="" THEN 170                  [574]
180 MODE 2:BORDER 1:INK 0,1:INK 1,26:PAPER [4127]
 0 :PEN 1
190 DIM feld$(10),feldlen(10),a$(10),k1(10 [5323]
),k2(10),satznum(22),y$(22)
200 WINDOW 1,80,7,23:WINDOW#1,1,80,1,3:WIN [5838]
DOW#2,1,80,25,25:PAPER#2,0:PEN#2,1:WINDOW#
3,1,80,24,24:WINDOW#4,1,80,5,6
210 '                                      [117]
220 '*****************                     [670]
230 '* PROGRAMMTITEL *                     [1582]
240 '*****************                     [670]
250 '                                      [117]
260 DEF FNfrsatz=INT((64000/geslen)-hsatz) [3495]
270 CLS#1:PRINT#1," ";STRING$(78,"*"):PRIN [8110]
T#1," ***           CPC 6128 - D A T E I
";CHR$(164);" by G. TREUSCH  Dez.1985 (V.1
.1)        ***":PRINT#1," ";STRING$(78,"*")
280  LOCATE 35,1:PRINT CHR$(18);CHR$(24);" [7414]
  .   .  ";CHR$(24);:LOCATE 1,1:PRINT CHR$(
7);:INPUT " Bitte Datum eingeben (TT.MM.JJ
): ",datum$
290 IF datum$="" THEN 280 ELSE IF MID$(dat [10913]
um$,3,1)<>"." OR MID$(datum$,6,1)<>"." OR
LEN(datum$)<>8 THEN LOCATE 1,3:PRINT" Datu
m bitte korrekt eingeben ! ":GOTO 280
300 '                                      [117]
310 '*************************             [768]
320 '* NEUE DATEI ERSTELLEN *              [773]
330 '*************************             [768]
340 '                                      [117]
350 CLS:PRINT CHR$(24);" Neue DATEI anlege [6003]
n (N), bestehende DATEI bearbeiten (B) ";C
HR$(24):opt$="nb":GOSUB 4140:IF k=2 THEN 6
20
360 CLS:PRINT CHR$(24);" Welchen Titel sol [14634]
l die DATEI erhalten? (max. 34 Zeichen) ";
CHR$(24):PRINT:PRINT">";SPACE$(34);"<":LOC
ATE 1,3:INPUT">",datei$:datei$=UPPER$(date
i$):ldatei=LEN(datei$):IF ldatei<1 THEN 36
0
370 PRINT:PRINT CHR$(24);" Wieviele Felder [11671]
 soll ein Datensatz enthalten ? (max. 15)
";CHR$(24);:INPUT" ",hfeld:IF hfeld<1 OR h
feld>15 THEN 370
380 PRINT:PRINT" Eingaben aendern (J/N)":o [5080]
pt$="jn":GOSUB 4140:IF k=1 THEN 360
390 ERASE feld$,feldlen,a$,k1,k2:DIM feld$ [6084]
(hfeld),feldlen(hfeld),a$(hfeld)
400 CLS:CLS#4:fetit1$="":fetit2$="":PRINT# [7733]
4,CHR$(24);" FELDNAME [Max. 20 Zeichen] F
ELDLAENGE [Min.2 (1 Feld),Max.255 (alle Fe
lder)] ";CHR$(24)
410 FOR i=1 TO hfeld:LOCATE 1,i:PRINT USIN [4826]
```

```
G"##";i;:PRINT" >";CHR$(24); SPACE$(20);CH
R$(24);"<":NEXT:PRINT CHR$(24)
420 FOR i=1 TO hfeld:LOCATE 5,i:INPUT"",fe [7895]
ld$(i):IF feld$(i)="" THEN 420 ELSE IF LEN
(feld$(i))>20 THEN feld$(i)=LEFT$(feld$(i)
,20)
430 feld$(i)=UPPER$(feld$(i)):fetit1$=feti [4970]
t1$+STR$(i)+"."+feld$(i):NEXT
440 maxlen=LEN(feld$(1)):FOR i=2 TO hfeld: [11112]
 maxlen=MAX(maxlen,LEN(feld$(i))):NEXT:max
len=maxlen+1:FOR i=1 TO hfeld:feld$(i)=fel
d$(i)+SPACE$(maxlen-LEN(feld$(i))):NEXT:PR
INT CHR$(24)
450 l1=78-maxlen                           [1816]
460 FOR i=1 TO hfeld:LOCATE 36,i:PRINT"(Ma [3808]
x.";l1;")"
470 LOCATE 32,i:PRINT CHR$(32)CHR$(8);:INP [4029]
UT"",feldlen(i)
475 IF feldlen(i)<2 THEN :GOTO 470         [1828]
480 geslen=geslen+feldlen(i)+1:IF geslen<= [3406]
255 THEN 510
490 LOCATE 32,i:PRINT CHR$(7);"Maximallaen [6161]
ge (255) ueberschritten !":pause=2000:GOSU
B 3920
500 FOR i=1 TO hfeld:LOCATE 32,i:PRINT CHR [2309]
$(18):NEXT: geslen=0:GOTO 460
510 NEXT i:geslen=geslen-1                 [1412]
520 fetit2$="     ":FOR i=1 TO hfeld:IF i<1 [4683]
0 THEN x=1 ELSE x=2
530 fetit2$=fetit2$+STR$(i)+SPACE$(feldlen [5117]
(i)-x):NEXT
540 PRINT:PRINT" Eingaben aendern (J/N)    [11137]
     ";CHR$(24);FNfrsatz;" freie Datensa
etze ";CHR$(24);:opt$="jn":GOSUB 4140:IF k
=1 THEN geslen=0:GOTO 400
550 GOSUB 560:GOTO 950                     [981]
560 CLS#4:CLS:PRINT CHR$(24);" Welchen Nam [12797]
en soll die DATEI erhalten? ";CHR$(24):LOC
ATE 42,1:INPUT"",file$:file$=LEFT$(file$,8
):file2$=file$+".str":file3$=file$+".dat":
file4$=file$+".dru":RETURN
570 '                                      [117]
580 '*****************                     [659]
590 '* DATEN EINLESEN *                    [722]
600 '*****************                     [659]
610 '                                      [117]
620 CLS:PRINT CHR$(24);CHR$(7);" Bitte geb [9405]
en Sie das aktuelle Laufwerk ein (A/B) ";C
HR$(24):opt$="ab":GOSUB 4140:IF k=1 THEN l
w$="A" ELSE lw$="B"
630 |DRIVE,lw$                             [1484]
640 CLS:PRINT CHR$(7);CHR$(24);" Welche DA [11968]
TEI moechten Sie bearbeiten ? ";CHR$(24):L
OCATE 42,1:INPUT"",file$:file$=LEFT$(file$
,8):file2$=file$+".str":file3$=file$+".dat
":file4$=file$+".dru"
650 GOSUB 710 :GOTO 950                    [873]
660 '                                      [117]
670 '**************************            [978]
680 '* DATEISTRUKTUR EINLESEN *            [1250]
690 '**************************            [978]
700 '                                      [117]
710 ON ERROR GOTO 730                      [1339]
720 CLS:PRINT CHR$(24);" DATEN werden eing [5244]
elesen, Moment bitte.....";CHR$(24):OPENIN
 file2$:GOTO 740
730 pause=3000:GOSUB 3920:RESUME 620       [2926]
740 ERASE feld$,feldlen,a$,k1,k2:geslen=0: [9021]
INPUT#9,hsatz,hfeld,datei$,maxlen,fetit1$,
fetit2$:ldatei=LEN(datei$)
750 DIM feld$(hfeld),feldlen(hfeld),a$(hfe [3563]
ld)
760 FOR i=1 TO hfeld:INPUT #9,feld$(i),fel [6447]
dlen(i):geslen=geslen+feldlen(i)+1:NEXT i:
geslen=geslen-1:CLOSEIN
770 |BANKOPEN,geslen:OPENIN file3$         [2817]
780 '                                      [117]
790 '*************************             [768]
800 '* DATENSAETZE EINLESEN *              [2233]
810 '*************************             [768]
820 '                                      [117]
830 r%=1:FOR i=1 TO hsatz:INPUT#9,satz$:LO [5140]
CATE 46,1:PRINT i
840 b$=STRING$(geslen," "):b$=satz$:|BANKW [4593]
RITE,@r%,b$,i
850 NEXT i:CLOSEIN                         [1222]
860 '                                      [117]
870 '********************************      [1223]
```



dieses Feldes wird dann entweder die ganze Datei oder ein wählbarer Ausschnitt der Datei auf- oder abwärts (ebenfalls wählbar) sortiert.

7. Programm beenden:
Dies ist der Ausstieg aus UNIDAT. Bevor jedoch das Programm beendet wird (und bei einem erneuten 'run' die Variablen gelöscht wären), wird abgefragt, ob die bestehende Datei abgespeichert werden soll.

8. Dateistruktur anzeigen:
Die Namen und Längen der benutzten Felder der aktuellen Datei werden ausgegeben, zusätzlich werden Informationen über die Anzahl der gespeicherten und noch freien Datensätze gegeben.

9. Anderes Programm starten:
Diese Option hat eine ähnliche Funktion wie Menuepunkt 7, zusätzlich wird die Möglichkeit geboten, nach Abspeichern der bearbeiteten Daten und Ausstieg aus UNIDAT, ein anderes Programm zu starten.

10. Disk-Katalog:
Hier kann jederzeit durch Tastendruck der CAT-Befehl ausgeführt werden, welcher über die auf der Arbeitsdiskette gespeicherten Dateien oder den noch verfügbaren Speicherplatz informiert.

Während der Arbeit mit UNIDAT wird der Benutzer in einer Kopfzeile ständig über den Namen und den Filenamen der gerade bearbeiteten Datei informiert. Beim Sprung vom Hauptmenue in ein Unterprogramm, erscheint der Name der aktuellen Funktion ebenfalls in der Kopfzeile.

Hier noch ein Tip zum Eintippen längerer Programme:

Speichern Sie in gewissen Zeitabständen Ihr Teilprogramm ab, um bei einem eventuellen Absturz oder Netzausfall nicht Stunden umsonst vor dem Bildschirm gesessen zu haben. Auch sollten Sie stets abspeichern, bevor Sie das Programm mit 'run' starten; unscheinbare Tippfehler können zum Verlust des Programms führen und Sie im Laufe der Zeit um Ihre Haarpracht bringen.

(ME)

```
880 '* DRUCKEREINSTELLUNG EINLESEN *            [853]
890 '******************************             [1223]
900 '                                           [117]
910 ON ERROR GOTO 930                           [1539]
920 OPENIN file4$:INPUT#9,zeiab$,zeil,leer [6383]
zei,schrift$,schrift,kompr:CLOSEIN:r%=0:RE
TURN
930 CLOSEIN:RETURN                          [651]
940 LOCATE#1,1,2:PRINT#1," ***  CPC 6128 - [7865]
 DATEI = ";datei$+SPACE$(36-ldatei);"File=
";UPPER$(file$);SPACE$(9-LEN(file$));"***
":RETURN
950 |BANKOPEN,geslen:DIM k1(hsatz),k2(hsat [4539]
z)
960 '                                       [117]
970 '**************                         [445]
980 '* HAUPTMENUE *                         [567]
990 '**************                         [445]
1000 '                                      [117]
1010 GOSUB 940: PAPER 0:PEN 1:CLS:PAPER#3, [4580]
0:CLS#3:PAPER#4,0:PEN#4,1:CLS#4:CLS#2
1020 LOCATE 26,1:PRINT"================== [9637]
======":LOCATE 26,2:PRINT"=  H A U P T M E
 N U E  =":LOCATE 26,3:PRINT"=============
============"
1030 ON BREAK CONT:ON ERROR GOTO 3550      [1966]
1040 DATA 1. DATEN eingeben,2. DATEN lesen [13133]
,3. DATEN abspeichern,4. DATEN einlesen,5.
 DATEN ausdrucken,6. DATEN ordnen,7. PROGR
AMM beenden,8. DATEISTRUKTUR anzeigen,9. a
nderes PROGRAMM starten,C. Disk-Katalog
1050 RESTORE 1040:FOR i=1 TO 10:READ menue [1624]
$(i)
1060 LOCATE 26,i+4:PRINT menue$(i):NEXT    [2047]
1070 LOCATE 29,i+5:PRINT "Bitte waehlen :" [4047]
:opt$="123456789C":GOSUB 4140
1080 CLS:LOCATE#1,24,2:PRINT#1,SPC(36):LOC [7511]
ATE#1,28,2:PRINT#1,menue$(k):ON k GOTO 159
0,1710,2550,1280,2970,2820,1210,1470,1100,
1350
1090 '                                      [117]
1100 '*****************************         [1164]
1110 '* ANDERES PROGRAMM STARTEN *          [1101]
1120 '*****************************         [1164]
1130 '                                      [117]
1140 CLS:LOCATE 20,8:PRINT CHR$(7);CHR$(24 [9149]
);" Haben Sie die Daten gesichert ? (J/N)
";CHR$(24):opt$="jn":GOSUB 4140:IF k=2 THE
N 1010
1150 CLS:CAT:PRINT CHR$(24);" Zurueck zum  [10163]
Hauptmenue [0] ";CHR$(24);:INPUT" welches
Programm moechten Sie starten?: ",ru$:IF r
u$="0" THEN 1010 ELSE RUN ru$
1160 '                                      [117]
1170 '*******************                   [665]
1180 '* PROGRAMM BEENDEN *                  [1251]
1190 '*******************                   [665]
1200 '                                      [117]
1210 LOCATE 6,8:PRINT CHR$(7);"P R O G R A [9727]
 M M - E N D E ";CHR$(24);" haben Sie die
Daten gesichert ? ";CHR$(24);" (J/N)":opt$
="jn":GOSUB 4140:ON k GOTO 1220,1010
1220 r%=0:FOR i=0 TO hsatz:b$="":|BANKWRIT [5821]
E,@r%,b$,i:NEXT:MODE 2:END
1230 '                                      [117]
1240 '*****************                     [659]
1250 '* DATEN EINLESEN *                    [722]
1260 '*****************                     [659]
1270 '                                      [117]
1280 LOCATE 6,8:PRINT CHR$(7);" D A T E I  [12010]
  l o e s c h e n ";CHR$(24);" haben Sie
die Daten gesichert ? ";CHR$(24);" (J/N)":
opt$="jn":GOSUB 4140:ON k GOTO 1290,1010
1290 r%=0:FOR i=0 TO hsatz:b$="":|BANKWRIT [5921]
E,@r%,b$,i:NEXT:GOSUB 620:GOTO 1010
1300 '                                      [117]
1310 '****************                      [528]
1320 '* DISC-KATALOG *                      [321]
1330 '****************                      [528]
1340 '                                      [117]
1350 GOSUB 1360:GOTO 1010                   [1920]
1360 CLS:CLS#2:CAT:PRINT#2,CHR$(24);" Moec [7971]
hten Sie eine Datei A)endern, L)oeschen od
er W)eiter ";CHR$(24):opt$="alw":GOSUB 414
0:ON k GOTO 1370,1400,1410
1370 CLS#2:PRINT#2,"|REN,"+CHR$(34)+"(neue [8847]
r Name)=";:PRINT#2,CHR$(24);SPACE$(8)+"."+
SPACE$(3):LOCATE#2,20,1:INPUT#2,"",neu$
1380 LOCATE#2,33,1:PRINT#2,CHR$(24);",(alt [6781]
er Name)=";CHR$(24);SPACE$(8)+"."+SPACE$(3
):LOCATE#2,47,1:INPUT#2,"",alt$
1390 |REN,neu$,alt$:PRINT#2,CHR$(24);:GOTO [2481]
 1360
1400 CLS#2:PRINT#2,"|ERA,"+CHR$(34);:INPUT [5460]
#2,"",x$:|ERA,x$:GOTO 1360
1410 CLS#2:RETURN                           [616]
1420 '                                      [117]
1430 '****************                      [670]
1440 '* DATEISTRUKTUR *                     [994]
1450 '****************                      [670]
1460 '                                      [117]
1470 satzlen=0:CLS:PRINT#4," Gespeicherte  [8524]
Datensaetze= ";hsatz;"    Freie Datensaetze
= ";FNfrsatz;
1480 PRINT#4,"    Datum ";datum$           [2567]
1490 ZONE 40:PRINT#4,CHR$(24);"     Feldname [4222]
:","Feldlaenge: ";CHR$(24)
1500 FOR i=1 TO hfeld:LOCATE 1,i:PRINT i;f [8092]
eld$(i):LOCATE 42,i:PRINT USING"##";feldle
n(i):satzlen=satzlen+feldlen(i):NEXT
1510 LOCATE 42,i:PRINT"---":LOCATE 29,i+1: [7532]
PRINT"Satzlaenge= ";:PRINT USING"###";satz
len;:PRINT" +";hfeld-1
1520 PRINT#2," Beliebige Taste druecken"   [2299]
1530 IF INKEY$="" THEN 1530 ELSE CLS#2:GOT [1997]
O 1010
1540 '                                      [117]
1550 '******************                    [659]
1560 '* DATEN EINGEBEN *                    [738]
1570 '******************                    [659]
1580 '                                      [117]
1590 CLS#2:PRINT#2,"           Eingabe mit [2936]
 [!] beenden"
1600 hsatz=hsatz+1:r%=hsatz:satz=hsatz     [3241]
1610 GOSUB 3750                             [1035]
1620 IF c$="!" THEN hsatz=hsatz-1:PRINT CH [6162]
R$(24):LOCATE 40,1:PRINT FNfrsatz;" Datens
aetze frei":GOTO 1640
1630 GOTO 1590                              [317]
1640 CLS#2:PRINT#2,"         W)eitere Einga [10221]
ben    A)enderungen     E)nde der Eingabe":
opt$="wae":GOSUB 4140:ON k GOTO 1590,1650,
1010
1650 GOSUB 3980:LOCATE 40,1:PRINT FNfrsatz [4826]
;" Datensaetze frei":GOTO 1640
1660 '                                      [117]
1670 '***************                       [345]
1680 '* DATEN LESEN *                       [565]
1690 '***************                       [345]
1700 '                                      [117]
1710 IF hsatz>0 THEN 1730                   [2412]
1720 CLS:LOCATE 20,8:PRINT CHR$(7);CHR$(24 [7736]
);" K E I N E   D A T E N   G E S P E I C
H E R T ";CHR$(24):pause=2000:GOSUB 3920:G
OTO 1010
1730 DATA 1. Datensaetze nach Nummern anze [9983]
igen,2. Datensaetze nach Kriterium suchen,
3. zurueck zum Hauptmenue
1740 RESTORE 1730:FOR i=1 TO 3:READ lese$: [4461]
LOCATE 26,2+(i*2):PRINT lese$:NEXT
1750 opt$="123":CLEAR INPUT:GOSUB 4140:CLS [3394]
:ON k GOTO 1760,2170,1010
1760 CLS:PRINT" Datensaetze  nach Nummern  [13861]
anzeigen ":PRINT:PRINT" 1. alle Datensetze
":PRINT:PRINT" 2. eine bestimmte Datensatz
-Nr.":PRINT:PRINT" 3. zurueck zum Hauptmen
ue"
1770 opt$="123":CLEAR INPUT:GOSUB 4140:le= [3636]
k:ON le GOTO 1950,1830,1010
1780 '                                      [117]
1790 '*********************                 [673]
1800 '* NACH NUMMERN LESEN *                [1702]
1810 '*********************                 [673]
1820 '                                      [117]
1830 CLS#2:PRINT#2,CHR$(7):INPUT#2," Welch [12414]
e Datensatz Nr. moechten Sie lesen? : ",sa
tz:IF satz<1 OR satz>hsatz THEN 1830
1840 CLS:PAPER#4,0:CLS#4:GOSUB 3760         [1308]
1850 LOCATE 2,1:PRINT"Datensatz-Nr.: ";sat [7466]
z;"       ":r%=satz:|BANKREAD,@r%,b$,satz
1860 PRINT CHR$(24);:LOCATE maxlen+2,3:a$( [5303]
1)=MID$(b$,1,feldlen(1)):PRINT a$(1)
1870 le=0:FOR zei=2 TO hfeld:le=le+feldlen [12203]
(zei-1):LOCATE maxlen+2,zei+2:a$(zei)=MID$
```

```
(b$,le+zei,feldlen(zei)):PRINT a$(zei):NEX
T:PRINT CHR$(24);
1880 PRINT#2," ";CHR$(240);CHR$(241);")Bla [9179]
ettern   S)uchen    A)endern    E)nde":opt$=
CHR$(240)+CHR$(241)+"sae":GOSUB 4140:ON k
GOTO 1890,1910,1940,1930,1010
1890 satz=satz+1:IF satz>hsatz THEN satz=1 [3617]
1900 GOTO 1850                             [337]
1910 satz=satz-1:IF satz<1 THEN satz=hsatz [3316]
1920 GOTO 1850                             [337]
1930 GOSUB 3980:LOCATE 40,1:PRINT FNfrsatz [4824]
;" Datensaetze frei":GOTO 1850
1940 CLS#2:PRINT#2,CHR$(7);" 1. Nummer suc [7576]
hen    2. Kriterium suchen":opt$="12":GOS
UB 4140:ON k GOTO 1830,2170
1950 satz=0:ze=0:po=1:CLS:PAPER#4,1:PEN#4, [8428]
0:CLS#4:PRINT#4,MID$(fetit1$,po,79):PRINT#
4,MID$(fetit2$,po,79)
1960 WHILE ze<17:satz=satz+1:r%=satz:ze=ze [4790]
+1:IF satz>hsatz THEN 2000
1970 |BANKREAD,@r%,b$,satz                 [2038]
1980 PRINT USING"####";satz;:PRINT " ";LOW [3255]
ER$(MID$(b$,po,74))
1990 WEND                                  [390]
2000 PRINT#2," ";CHR$(240);")Weiter   ";CHR [8043]
$(242);")nach Links  ";CHR$(243);")nach Re
chts  ";" A)aendern   S)uchen   E)beenden"
2010 opt$=CHR$(240)+CHR$(242)+CHR$(243)+"a [5610]
se":GOSUB 4140:ON k GOTO 2020,2040,2070,21
00,1940,1010
2020 ze=0:satz=satz-1:IF satz<0 THEN satz= [5381]
0 ELSE IF satz+1>hsatz THEN 2110
2030 CLS:CLS#2::GOTO 1960                  [776]
2040 satz=satz-ze:IF satz<0 THEN satz=0    [2945]
2050 ze=0:po=po+40:IF po>geslen THEN po=1  [2806]
2060 CLS:CLS#2:CLS#4:PRINT#4,MID$(fetit1$, [3843]
po,79):PRINT#4,MID$(fetit2$,po,79):GOTO 19
60
2070 satz=satz-ze:IF satz<0 THEN satz=0    [2945]
2080 ze=0:po=po-40:IF po<1 THEN po=1       [2425]
2090 CLS:CLS#2:CLS#4:PRINT#4,MID$(fetit1$, [3843]
po,79):PRINT#4,MID$(fetit2$,po,79):GOTO 19
60
2100 GOSUB 3980:GOTO 2000                  [1558]
2110  CLS#2:PRINT#2:PRINT#2,CHR$(7);"      [6632]
   ";CHR$(24);" D A T E I E N D E ";CH
R$(24);:pause=2000:GOSUB 3920:GOTO 1010
2120 '                                     [117]
2130 '************************             [768]
2140 '* NACH KRITERIUM LESEN *             [1058]
2150 '************************             [768]
2160 '                                     [117]
2170 CLS:CLS#2:PAPER#4,0:PEN#4,1:CLS#4:PRI [7214]
NT CHR$(24);" Feldbezeichnung ":PRINT CHR$
(24)
2180 ZONE 26:FOR i=1 TO hfeld:PRINT USING" [3941]
##";i;:PRINT ". ";feld$(i),:NEXT
2190 LOCATE 17,13:PRINT CHR$(18);CHR$(7);C [11066]
HR$(24);:INPUT" In welchem Feld moechten S
ie DATEN suchen? Nr. ",feld:IF feld<1 OR f
eld>hfeld THEN 2190
2200 GOSUB 2210:GOTO 2240                  [1582]
2210 x=0:IF feld=1 THEN 2230               [2461]
2220 FOR i=2 TO feld:x=x+feldlen(i-1):NEXT [3898]
2230 x=x+feld:RETURN                       [1540]
2240 r%=1:LOCATE 17,13:PRINT CHR$(7);"     [13334]
```

```
     Bitte SUCH-KRITERIUM eingeben
   ";CHR$(24):PRINT:PRINT " ";feld$(feld)
:LOCATE 23,15:PRINT"(als Universalzeichen
kann [?] eingegeben werden)"
2250 PRINT:PRINT " >";CHR$(24);SPACE$(feld [3705]
len(feld));CHR$(24);"<"
2260 LOCATE 3,17:INPUT"",su$:IF su$="" THE [8186]
N 1010 ELSE IF LEN(su$)>feldlen(feld) THEN
 su$=LEFT$(su$,feldlen(feld))
2270 FOR y=1 TO LEN(su$):IF MID$(su$,y,1)= [5376]
"?" THEN MID$(su$,y,1)=CHR$(0)
2280 NEXT                                  [350]
2290 su$=STRING$(x-1,0)+su$                [2098]
2300 ze=0:po=1:CLS:PAPER#4,1:PEN#4,0:CLS#4 [6508]
:PRINT#4,MID$(fetit1$,po,79):PRINT#4,MID$(
fetit2$,po,79)
2310 |BANKFIND,@r%,LOWER$(su$),r%:|BANKREA [4303]
D,@r%,b$
2320 IF r%<0 OR r%>hsatz THEN 2360         [1399]
2330 ze=ze+1:IF ze=18 THEN 2370 ELSE j=ze  [2400]
2340 PRINT USING"####";r%;:PRINT " ";LOWER [6246]
$(MID$(b$,po,74)):satznum(ze)=r%:y$(ze)=b$
2350 r%=r%+1:GOTO 2310                     [1586]
2360 PRINT#2,CHR$(7);" Suchkriterium: ";CH [7765]
R$(24);" ";LOWER$(su$);" ";CHR$(24);" nich
t weiter vorhanden ! ":pause= 3000:GOSUB 3
920
2370 PRINT#2," ";CHR$(240);")Weiter   ";CHR [8043]
$(242);")nach Links  ";CHR$(243);")nach Re
chts  ";" A)aendern   S)uchen   E)beenden"
2380 opt$=CHR$(240)+CHR$(242)+CHR$(243)+"a [4929]
se":GOSUB 4140:ON k GOTO 2390,2460,2420,24
10,1940,1010
2390 CLS#2:ze=0:r%=r%-1:IF r%<0 OR r%>hsat [3948]
z THEN 2110
2400 CLS:GOTO 2310                         [1044]
2410 GOSUB 3980:GOTO 2370                  [1620]
2420 '                                     [117]
2430 ze=0:po=po-40:IF po<1 THEN po=1       [2425]
2440 CLS#4:PRINT#4,MID$(fetit1$,po,79):PRI [3832]
NT#4,MID$(fetit2$,po,79)
2450 CLS:CLS#2:FOR i=1 TO j:PRINT USING"## [5393]
##";satznum(i);:PRINT " ";LOWER$(MID$(y$(i
),po,74)):NEXT:GOTO 2370
2460 '                                     [117]
2470 po=po+40:IF po>geslen THEN po=1       [1864]
2480 CLS#4:PRINT#4,MID$(fetit1$,po,79):PRI [3832]
NT#4,MID$(fetit2$,po,79)
2490 CLS:CLS#2:FOR i=1 TO j:PRINT USING"## [5393]
##";satznum(i);:PRINT " ";LOWER$(MID$(y$(i
),po,74)):NEXT:GOTO 2370
2500 '                                     [117]
2510 '******************                   [669]
2520 '* DATEN SPEICHERN *                  [1890]
2530 '******************                   [669]
2540 '                                     [117]
2550 CLS:PRINT CHR$(24);CHR$(7);" DATEN ab [7489]
speichern (J/N) ";CHR$(24):opt$="jn":GOSUB
 4140:IF k=2 THEN 1010
2560 CLS:PRINT CHR$(24);CHR$(7);" Bitte ge [9405]
ben Sie das aktuelle Laufwerk ein (A/B) ";
CHR$(24):opt$="ab":GOSUB 4140:IF k=1 THEN
lw$="A" ELSE lw$="B"
2570 |DRIVE,lw$                            [1484]
2580  CLS:PRINT CHR$(24);CHR$(7);" Moechte [10717]
n Sie die Diskette auf freien Speicherplat
```

```
z ueberpruefen (J/N) ";CHR$(24):opt$="jn":
GOSUB 4140:IF k=1 THEN GOSUB 1360
2590 CLS:PRINT CHR$(24);CHR$(7);" Moechten [8396]
 Sie den Dateinamen aendern (J/N) ";CHR$(2
4):opt$="jn":GOSUB 4140:IF k=1 THEN GOSUB
560
2600 '                                   [117]
2610 '****************************       [1051]
2620 '* DATEISTRUKTUR SPEICHERN *        [1588]
2630 '****************************       [1051]
2640 '                                   [117]
2650 CLS:PRINT CHR$(24);" DATEN werden ges [4665]
peichert, Moment bitte..... ";CHR$(24):OPE
NOUT file2$
2660 WRITE#9,hsatz,hfeld,datei$,maxlen,fet [3590]
it1$,fetit2$
2670 FOR i=1 TO hfeld:WRITE#9,feld$(i),fel [5534]
dlen(i):NEXT:CLOSEOUT
2680 '                                   [117]
2690 '***************************        [894]
2700 '* DATENSAETZE SPEICHERN *          [3102]
2710 '***************************        [894]
2720 '                                   [117]
2730 OPENOUT file3$                      [1382]
2740 r%=1:FOR i=1 TO hsatz:|BANKREAD,@r%,b [8473]
$,i:satz$=STRING$(geslen," "):satz$=b$:LOC
ATE 46,1:PRINT i
2750 WRITE#9,satz$                       [1684]
2760 NEXT i:CLOSEOUT:r%=0:GOTO 1010      [1625]
2780 '****************                   [528]
2790 '* DATEN ORDNEN *                   [987]
2800 '****************                   [528]
2810 '                                   [117]
2820 IF hsatz=0 THEN 1720                [2441]
2830 CLS:CLS#2:ZONE 26:FOR i=1 TO hfeld:PR [5288]
INT USING"##";i;:PRINT ". ";feld$(i),:NEXT
2840 LOCATE 17,15:PRINT CHR$(7);CHR$(24);" [10564]
 Welches Feld moechten Sie ordnen ? Nr.: "
;CHR$(24):LOCATE 60,15:PAPER 0:PEN 1:INPUT
"",feld:IF feld<1 OR feld>hfeld THEN 2840
2850 GOSUB 2210:l=feldlen(feld)          [1730]
2860 PRINT:PRINT CHR$(24);"   1. gesamte D [8235]
ATEI ordnen, 2. Teil der DATEI ordnen ":
opt$="12":GOSUB 4140:IF k=1 THEN 2880
2870 GOSUB 3640:f=lnum:g=hnum:GOTO 2890  [2663]
2880 f=1:g=hsatz                         [1628]
2890 CLS#2:PRINT#2,CHR$(24);" 1. aufwaerts [8211]
 ordnen   2. abwaerts ordnen ";CHR$(24):op
t$="12":GOSUB 4140:sort=k
2900 CLS#2:PRINT#2,CHR$(24);" O R D N U N  [4669]
G laeuft, bitte warten... ";CHR$(24)
2910 PRINT CHR$(7);:GOSUB 4220:PRINT CHR$( [2162]
7);CHR$(24);:GOTO 1010
2930 '*******************                [665]
2940 '* DATEN AUSDRUCKEN *               [819]
2950 '*******************                [665]
2960 '                                   [117]
2970 IF hsatz=0 THEN 1720                [2441]
2980 CLS:PRINT CHR$(7);CHR$(24);" Daten au [5494]
sdrucken (J/N) ";CHR$(24):opt$="jn":GOSUB
4140:IF k=2 THEN 1010
2990 CLS:PRINT CHR$(7);CHR$(24);" Moechten [9029]
 Sie die Druckereinstellung aendern (J/N)
";CHR$(24):opt$="jn":GOSUB 4140:IF k=1 THE
N GOSUB 4580
3000 CLS:PRINT CHR$(7);CHR$(24);" Bitte Dr [7868]
ucker einschalten und beliebige Taste drue
cken ";CHR$(24)
3010 IF INKEY$="" THEN 3010              [798]
3020 PRINT#8,CHR$(27);zeiab$;CHR$(27);"C"; [11298]
CHR$(zeil);CHR$(27);"N";CHR$(leerzei);CHR$
(27);schrift$;CHR$(27);"W";CHR$(schrift);C
HR$(27);CHR$(kompr);
3030 CLS:PRINT" 1. Datensaetze ausdrucken" [10504]
:PRINT:PRINT" 2. Adressenaufkleber ausdruc
ken":PRINT:PRINT" 3. zurueck zum Hauptmenu
e"
3040 opt$="123":GOSUB 4140:ON k GOTO 3050, [2918]
3280,1010
3050 CLS:PRINT CHR$(24);" A)lle Datensaetz [9362]
e ausdrucken  N)ach Nummern ausdrucken ";C
HR$(24):opt$="an":GOSUB 4140:IF k=1 THEN 3
070
3060 GOSUB 3640:CLS:PRINT CHR$(24);" Mocht [8978]
en Sie den Kopf mit ausdrucken (J/N) ";CHR
$(24):opt$="jn":GOSUB 4140:kopf=k:GOTO 308
0
3070 kopf=1:lnum=1:hnum=hsatz            [1261]
3080 CLS:PRINT CHR$(24);" Soll nach jedem [11060]
Datensatz eine Leerzeile gedruckt werden (
J/N) ";CHR$(24):opt$="jn":GOSUB 4140:leer=
k
3090 IF kopf=2 THEN 3160                 [1627]
3100 IF kompr=15 AND (geslen+5)<=132 THEN [7608]
br=geslen+5 ELSE IF kompr=15 AND (geslen+5
)>132 THEN br=132
3110 IF kompr=18 AND (geslen+5)<=80 THEN b [6046]
r=geslen+5 ELSE IF kompr=18 AND (geslen+5)
>80 THEN br=80
3120 PRINT#8,CHR$(27);"D";CHR$(br-15);CHR$ [2320]
(0);
3130 PRINT#8,CHR$(27);"G";STRING$(br,"*") [2252]
3140 PRINT#8,CHR$(27);"W";CHR$(1);CHR$(27) [7841]
;"-";CHR$(1);datei$;CHR$(27);"-";CHR$(0);C
HR$(27);"W";CHR$(0);CHR$(9);"Stand: ";datu
m$
3150 PRINT#8:PRINT#8,fetit1$:PRINT#8,fetit [7911]
2$:PRINT#8,STRING$(br,"*");CHR$(27);"H":PR
INT#8
3160 CLS:FOR i=lnum TO hnum:r%=i         [2021]
3170 |BANKREAD,@r%,b$,i                  [1510]
3180 PRINT b$                            [476]
3190 PRINT#8,USING"####";i;:PRINT#8," ";UP [3369]
PER$(b$)
3200 IF leer=2 THEN 3220                 [693]
3210 PRINT#8,STRING$(br,"-")             [1553]
3220 NEXT i:GOTO 1010                    [856]
3240 '******************************     [1143]
3250 '* ADRESSENAUFKLEBER DRUCKEN *      [1018]
3260 '******************************     [1143]
3280 PRINT#8,CHR$(27);"O";CHR$(27);"P";CHR [2417]
$(18);
3290 CLS:PRINT CHR$(24);" F e l d b e z e [9021]
i c h n u n g ";CHR$(24):PRINT:ZONE 26:FOR
 i=1 TO hfeld:PRINT USING"##";i;:PRINT ".
";feld$(i),:NEXT
3300 LOCATE 10,11:PRINT CHR$(24);" Format [7708]
";CHR$(24);:PRINT"      (fuer Code-Nr. kann
[0] oder [ENTER] eingegeben werden)"
3310 PRINT:PRINT CHR$(24);"               [9318]
 Code-Nr.  ":PRINT"    N  a  m  e
   ":PRINT"   S  t  r  a  s  s  e    ":PRI
NT:PRINT"    Plz ";CHR$(24);" ";CHR$(24)"
 O   r   t       ";CHR$(24)
3320 FOR j=1 TO 3:LOCATE 29,j+12:PRINT CHR [7112]
$(242);:INPUT" Feld Nr.: ",feld:format(j)=
feld:GOSUB 2210:po(j)=x:NEXT
3330 LOCATE 29,17:PRINT CHR$(242);:INPUT" [10327]
Feld Nr.: ",feld:format(4)=feld:GOSUB 2210
:po(4)=x:LOCATE 45,17:INPUT"Feld Nr.: ",fe
ld:format(5)=feld:GOSUB 2210:po(5)=x
3340 CLS:PRINT" 1. Aufkleber einbahnig  2 [7702]
. Aufkleber zweibahnig  ausdrucken":opt$="
12":GOSUB 4140:st=k
3350 CLS:PRINT" 1. alle Adressen  2. Adres [6521]
sen nach Nr. ausdrucken":opt$="12":GOSUB 4
140:ON k GOTO 3370,3360
3360 GOSUB 3640:GOTO 3380                [1620]
3370 lnum=1:hnum=hsatz                   [2037]
3380 CLS:FOR i=lnum TO hnum:j=i+1:IF j>hnu [2753]
m THEN j=i
3390 IF po(1)=0 THEN po(1)=1:MID$(b$,po(1) [4890]
,feldlen(format(1)))=" "
3400 r%=i:|BANKREAD,@r%,b$,i:PRINT#8:PRINT [7427]
#8,TAB(24);MID$(b$,po(1),feldlen(format(1)
));:IF st=1 THEN 3420
3410 r%=j:|BANKREAD,@r%,b$,j:PRINT#8,TAB(6 [7996]
2);MID$(b$,po(1),feldlen(format(1)));:r%=i
:|BANKREAD,@r%,b$,i
3420 PRINT#8:PRINT#8:PRINT#8,TAB(4);UPPER$ [6310]
(MID$(b$,po(2),feldlen(format(2))));:IF st
=1 THEN 3440
3430 r%=j:|BANKREAD,@r%,b$,j:PRINT#8,TAB(4 [8155]
2);UPPER$(MID$(b$,po(2),feldlen(format(2))
));:r%=i:|BANKREAD,@r%,b$,i
3440 PRINT#8:PRINT#8,TAB(4);UPPER$(MID$(b$ [7250]
,po(3),feldlen(format(3))));:IF st=1 THEN
3460
3450 r%=j:|BANKREAD,@r%,b$,j:PRINT#8,TAB(4 [9688]
2);UPPER$(MID$(b$,po(3),feldlen(format(3))
));:r%=i:|BANKREAD,@r%,b$,i
3460 PRINT#8:PRINT#8:PRINT#8,TAB(4);UPPER$ [10251]
(MID$(b$,po(4),feldlen(format(4))));:PRINT
#8,TAB(9);UPPER$(MID$(b$,po(5),feldlen(for
mat(5))));:IF st=1 THEN 3480
3470 r%=j:|BANKREAD,@r%,b$,j:PRINT#8,TAB(4 [11388]
2);UPPER$(MID$(b$,po(4),feldlen(format(4))
));:PRINT#8,TAB(47);UPPER$(MID$(b$,po(5),f
```

```
eldlen(format(5))));:i=i+1
3480 PRINT#8:PRINT#8:PRINT#8:PRINT#8:PRINT  [3351]
#8:PRINT#8:NEXT i
3490 GOTO 1010                               [345]
3500 '                                       [117]
3510 '****************                       [670]
3520 '* ERROR ROUTINE *                      [1778]
3530 '****************                       [670]
3540 '                                       [117]
3550 CLS:PRINT,CHR$(7);" Fehler";ERR;"in Z  [2835]
eile";ERL
3560 PRINT:PRINT," beliebige Taste druecke  [4010]
n"
3570 ON BREAK STOP                           [431]
3580 IF INKEY$="" THEN 3580 ELSE RESUME 10  [1887]
10
3600 '**************                         [345]
3610 '* NR.-ROUTINE *                        [524]
3620 '**************                         [345]
3630 '                                       [117]
3640 CLS#2:PRINT#2,CHR$(24);" Mit [0] zuru  [8426]
eck zum Hauptmenue ";CHR$(24);:INPUT#2,"
von Nr.: ",lnum:IF lnum<1 THEN 1010
3650 IF lnum>hsatz THEN 3640                 [979]
3660 LOCATE#2,53,1:INPUT#2,"bis Nr.: ",hnu  [4117]
m:IF hnum=0 THEN 1010
3670 IF hnum<lnum THEN 3660                  [1828]
3680 IF hnum>hsatz THEN hnum=hsatz           [3344]
3690 RETURN                                  [555]
3710 '*****************                      [659]
3720 '* EINGABEROUTINE *                     [1219]
3730 '*****************                      [659]
3740 '                                       [117]
3750 GOSUB 3760:GOTO 3780                    [1562]
3760 LOCATE 2,1:PRINT"Datensatz-Nr.: ";sat  [9323]
z:LOCATE 40,1:PRINT FNfrsatz;" Datensaetze
 frei"
3770 FOR i=1 TO hfeld:LOCATE 1,i+2:PRINT f  [10082]
eld$(i);">";CHR$(24);SPACE$(feldlen(i));CH
R$(24);"<";CHR$(18):NEXT:RETURN
3780 FOR zei=1 TO hfeld:PRINT CHR$(24);:LO  [5274]
CATE maxlen+2,zei+2:INPUT"",c$:IF c$="!" T
HEN RETURN
3790 IF c$="" THEN a$(zei)=a$(zei) ELSE a$  [3651]
(zei)=c$
3800 IF LEN(a$(zei))>feldlen(zei) THEN a$(  [3843]
zei)=LEFT$(a$(zei),feldlen(zei))
3810 IF LEN(a$(zei))<feldlen(zei) THEN a$(  [5434]
zei)=a$(zei)+STRING$(feldlen(zei)-LEN(a$(z
ei)),"  ")
3820 PRINT CHR$(24);:NEXT zei                [1939]
3830 c$=""                                   [431]
3840 FOR i=1 TO hfeld-1:c$=c$+a$(i)+" ":NE  [3924]
XT:c$=c$+a$(hfeld):b$=c$
3850 FOR i=1 TO hfeld:a$(i)="":NEXT          [1065]
3860 |BANKWRITE,@r%,b$,satz:RETURN           [3790]
3870 '                                       [117]
3880 '****************                       [670]
3890 '* PAUSENROUTINE *                      [1684]
3900 '****************                       [670]
3910 '                                       [117]
3920 FOR i=1 TO pause:NEXT:RETURN            [2014]
3930 '                                       [117]
3940 '*******************                    [657]
3950 '* DATENSATZ AENDERN *                  [994]
3960 '*******************                    [657]
3970 '                                       [117]
3980 CLS#2:PRINT#2," 1. Datensatz aendern   [6747]
  2. Datensatz loeschen   3. Hauptmenue":o
pt$="123":GOSUB 4140:ON k GOTO 4020,3990,1
010
3990 CLS#2:INPUT#2," Welchen Datensatz moe  [9246]
chten Sie loeschen ? Nr.: ",satz:LOCATE#2,
50,1::PRINT#2,"   Moment bitte..."
4000 FOR i=satz TO hsatz:r%=i:|BANKREAD,@r  [8461]
%+1,b$,i+1:|BANKWRITE,@r%,b$,i:NEXT i:hsat
z=hsatz-1
4010 r%=0:CLS#2:PRINT#2,"  W)eitere Datens  [8632]
aetze loeschen     E)nde":opt$="we":GOSUB 4
140:IF k=1 THEN 3990 ELSE RETURN
4020 PAPER#3,0:PEN#3,1:CLS#3:CLS#2:INPUT#2  [10992]
," Welchen Datensatz moechten Sie aendern
? Nr.: ",satz:IF satz<1 OR satz>hsatz  THE
N 4020
4030 r%=satz:|BANKREAD,@r%,b$,satz           [1932]
4040 CLS:CLS#2:GOSUB 3760:PRINT CHR$(24);:  [4766]
LOCATE maxlen+2,3:a$(1)=MID$(b$,1,feldlen(
1)):PRINT a$(1)
4050 le=0:FOR zei=2 TO hfeld:le=le+feldlen  [11699]
(zei-1):LOCATE maxlen+2,zei+2:a$(zei)=MID$
(b$,le+zei,feldlen(zei)):PRINT a$(zei):NEX
T
4060 PRINT CHR$(24);:CLS#2:PRINT#2,CHR$(24  [10612]
);" Wenn Sie ein Feld nicht aendern moecht
en, [ENTER] oder [RETURN] druecken. ";CHR$
(24)
4070 GOSUB 3780:CLS#2:PRINT#2," Moechten S  [8610]
ie weitere Datensaetze aendern ? (J/N)":op
t$="jn":GOSUB 4140:IF k=1 THEN 4020
4080 PAPER#3,0:PEN#3,1:CLS#3:RETURN          [2395]
4100 '***************                        [528]
4110 '* INKEYROUTINE *                       [996]
4120 '***************                        [528]
4130 '                                       [117]
4140 k$=INKEY$:IF k$="" THEN 4140            [2964]
4150 IF INSTR(UPPER$(opt$),UPPER$(k$))=0 T  [2651]
HEN 4140
4160 k=INSTR(UPPER$(opt$),UPPER$(k$)):RETU  [3287]
RN
4170 '                                       [117]
4180 '************                           [483]
4190 '* QUICKSORT *                          [1027]
4200 '************                           [483]
4210 '                                       [117]
4220 ERASE k1,k2:DIM k1(hsatz),k2(hsatz):n  [3316]
1=f:i=f:n2=g
4230 j1=n1:j2=n2                             [1392]
4240 r%=j1:|BANKREAD,@r%,b$,j1:c$=b$         [1440]
4250 r%=j2:|BANKREAD,@r%,b$,j2:d$=b$         [2031]
4260 IF sort=2 THEN 4280                     [691]
4270 IF UPPER$(MID$(c$,x,l))<=UPPER$(MID$(  [3138]
d$,x,l)) THEN 4420 ELSE 4290
4280 IF UPPER$(MID$(c$,x,l))>=UPPER$(MID$(  [3492]
d$,x,l)) THEN 4420
4290 x$=c$:c$=d$:d$=x$                       [721]
4300 b$=d$:r%=j2:|BANKWRITE,@r%,b$,j2        [2131]
4310 b$=c$:r%=j1:|BANKWRITE,@r%,b$,j1        [2090]
4320 j1=j1+1                                 [908]
4330 IF j1=j2 THEN 4440                      [1217]
4340 r%=j1:|BANKREAD,@r%,b$,j1:c$=b$         [1440]
4350 r%=j2:|BANKREAD,@r%,b$,j2:d$=b$         [2031]
4360 IF sort=2 THEN 4380                     [751]
4370 IF UPPER$(MID$(c$,x,l))<=UPPER$(MID$(  [3588]
d$,x,l)) THEN 4320 ELSE 4390
4380 IF UPPER$(MID$(c$,x,l))>=UPPER$(MID$(  [3273]
d$,x,l)) THEN 4320
4390 x$=c$:c$=d$:d$=x$                       [721]
4400 b$=d$:r%=j2:|BANKWRITE,@r%,b$,j2        [2131]
4410 b$=c$:r%=j1:|BANKWRITE,@r%,b$,j1        [2090]
4420 j2=j2-1                                 [568]
4430 IF j2<>j1 THEN 4240                     [1199]
4440 j2=j2+1                                 [572]
4450 IF j2>=n2 THEN 4480                     [1538]
4460 k1(i)=j2:k2(i)=n2                       [1736]
4470 i=i+1                                   [307]
4480 j1=j1-1                                 [872]
4490 IF n1<j1 THEN n2=j1:GOTO 4230           [2035]
4500 i=i-1                                   [319]
4510 n1=k1(i):n2=k2(i):IF i>f-1 THEN 4230    [2103]
4520 RETURN                                  [555]
4540 '**********************************     [1460]
4550 '* DRUCKEREINSTELLUNG (Riteman F+) *    [1827]
4560 '**********************************     [1460]
4570 '                                       [117]
4580 CLS:PRINT" Welchen Zeilenabstand moec  [9289]
hten Sie einstellen ?":PRINT:INPUT " 1/8 Z
oll=0     1/6 Zoll=2    ",zeiab$
4590 CLS:INPUT " Wieviel Zeilen moechten S  [5670]
ie pro Seite drucken ?: ",zeil
4600 PRINT:INPUT " Wieviel Leerzeilen moec  [8042]
hten Sie nach jeder Seite drucken ?: ",lee
rzei:zeil=zeil+leerzei '( P E R F O R A T
I O N   U E B E R S P R I N G E N )
4610 CLS:PRINT" In welcher Schriftart moec  [10511]
hten Sie drucken ?  ":PRINT:INPUT " Pica=P
   Elite=M   Italic=4  ",schrift$:schrift$
=UPPER$(schrift$)
4620 PRINT:PRINT" In welcher Schriftbreite  [8360]
 moechten Sie drucken ?  ":INPUT " Normal=
0   Breit=1    ",schrift
4630 PRINT:PRINT " Moechten Sie in komprim  [10747]
ierter Schrift drucken (J/N)":opt$="jn":GO
SUB 4140:IF k=1 THEN kompr=15 ELSE kompr=1
8
4640 OPENOUT file4$                          [1420]
4650 WRITE#9,zeiab$,zeil,leerzei,schrift$,  [4942]
schrift,kompr
4660 CLOSEOUT:RETURN                         [1447]
```

CPC 464 als Turn-Key-System

Mit einem kleinen Schnitt, 10 cm Flachbandkabel (zwei Adern) und einem Ausschalter ist es möglich, bei jedem Einschalten und jedem RESET sofort in das CP/M "durchzustarten".

Dazu muß lediglich auf der Komponentenseite (Oberseite) des Schneider Floppy-Controllers die Lötbrücke LK1 mit einem scharfen Messer durchgetrennt werden.

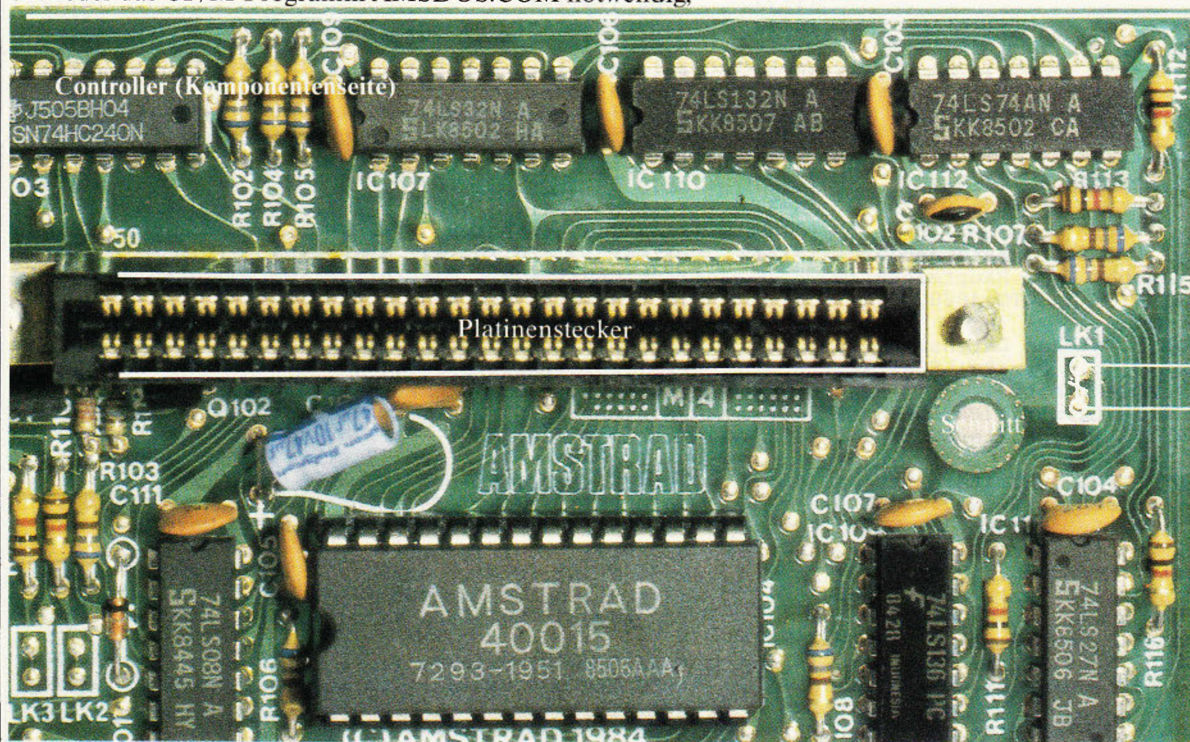
Nach Unterbrechung dieser Lötbrücke versucht das AMSDOS bei jeder Initialisierung (Einschalten/Reset) CP/M von Diskette zu laden und zu starten.

Um den CPC unter dem Locomotive-Basic zu betreiben, ist entweder das CP/M-Programm AMSDOS.COM notwendig, oder eben der Schalter, mit dem das "Turn-Key-System" einfach wieder abgeschaltet wird (indem die Brücke wieder geschlossen wird), so daß der Schneider wieder ganz normal arbeitet.

Um den Ausschalter zu nutzen, wird an den Lötpunkten der Lötbrücke je eine Ader des Flachbandkabels angelötet; die anderen Kabelenden werden am Schalter angelötet, der bei entsprechender Größe leicht im Controller-Gehäuse untergebracht werden kann.

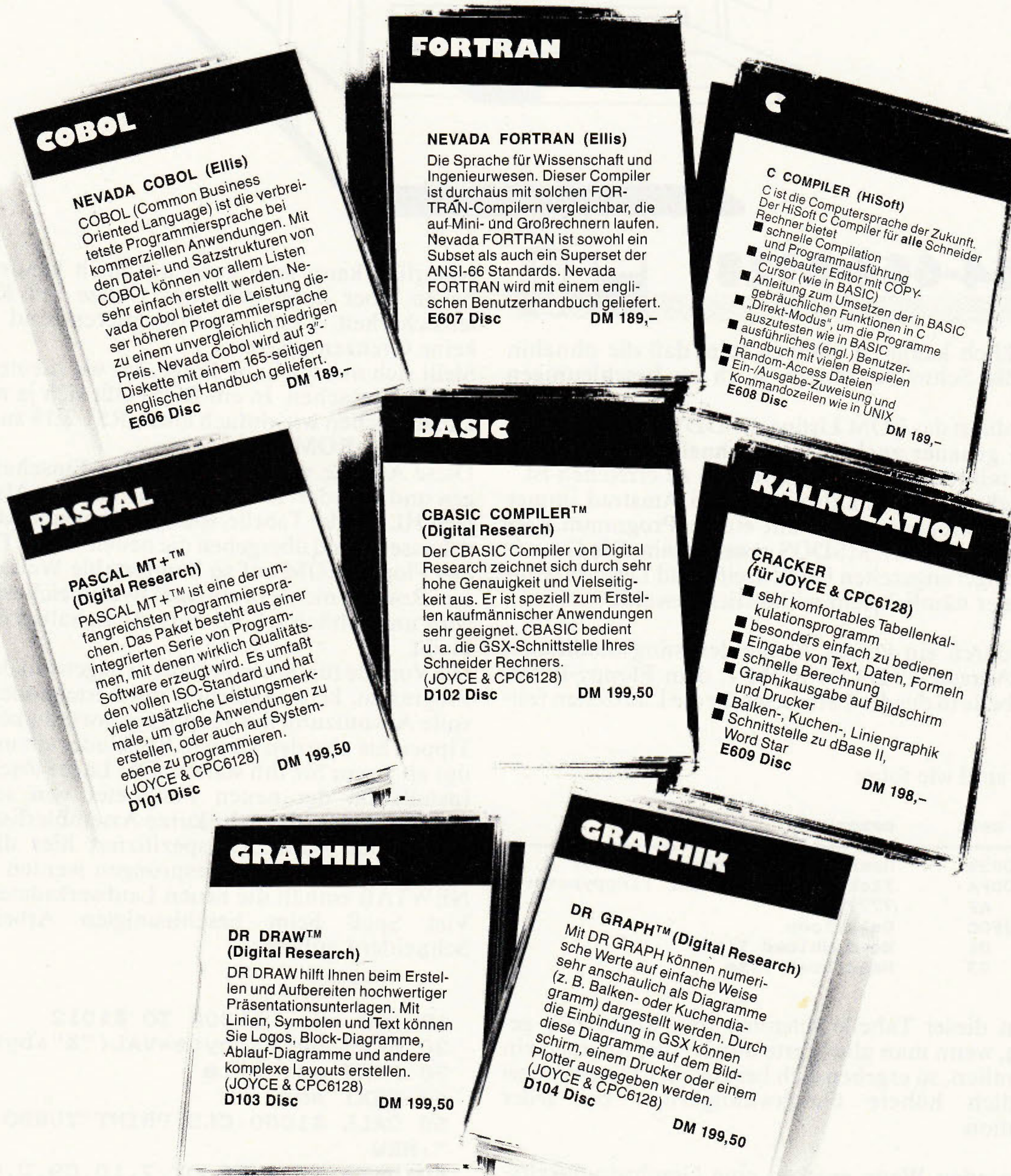
Aber Achtung! Nach diesem Hardwareeingriff übernimmt die Firma Schneider mit großer Wahrscheinlichkeit keine Garantie mehr.

Arnold Bemberg



SchneiderData

Profi-Software für Schneider

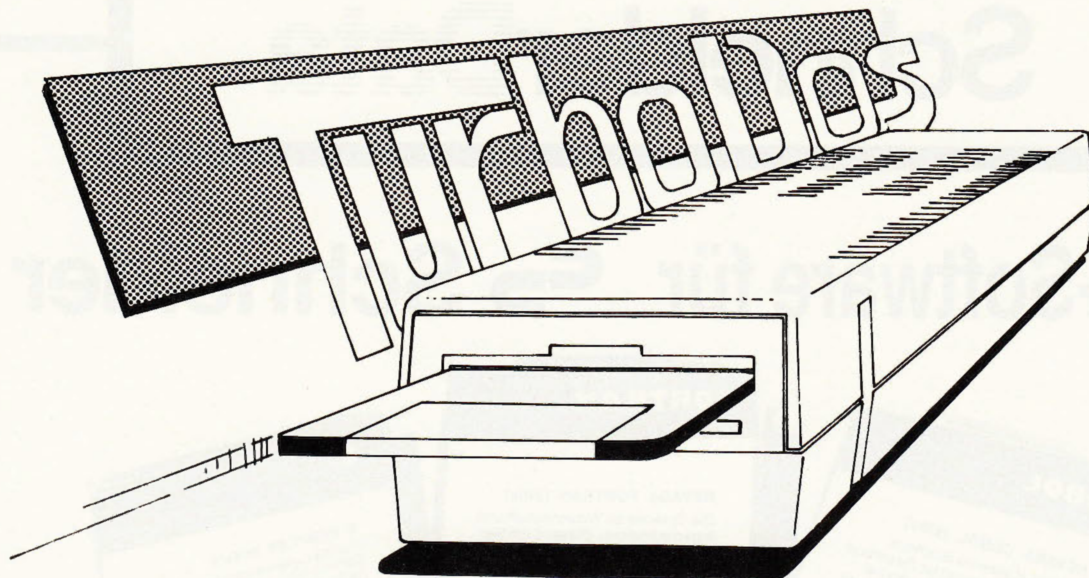


Computerspiele & Zubehör für die CPC-Familie

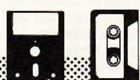
	Cass.	Disc.
3D STUNT RIDER (Motorrad Stuntfahren)	29,50	49,—
SOCCER (Fußball)	39,50	59,—
3D INVADERS (Weltraumspiel)	29,50	49,—
ROLAND AHOY!	39,50	59,—
AMERICAN FOOTBALL	39,50	59,—
CENTRE COURT (Tennis)	29,50	49,—
TRAFFIC (Verkehrsregelung)	39,50	59,—
FRUIT MACHINE (Spielautomat)	29,50	49,—
CLASSIC RACING (Pferderennen)	29,50	49,—
GRAND PRIX RALLY II (Autorennen)	39,50	59,—
ASTRO ATTACK (Weltraumspiel)	29,50	49,—
LORDS OF MIDNIGHT (Adventure)	39,50	59,—
TOMBESTOWN	39,50	59,—
3D BOXING (Amsoft Gold)	39,50	59,—
MACROCOSMICA (Amsoft Gold)	39,50	59,—
ALEX HIGGINS POOL (Poolbillard)	39,50	59,—

LIGHTPEN LP-1 mit Software	99,—
CPC-PIO (zum Steuern & Regeln)	98,—
CPC-UHR (Selbstbausatz)	98,—
SPRACHSYNTHESIZER SSA-1	148,—
CP/M 2.2 USER GUIDE (orig. engl.)	59,—
GUIDE TO LOGO (orig. engl.)	69,—
DISKETTENBOX für 15 3" Disketten	39,—
DISKETTENBOX für 30 3" Disketten	49,—

SchneiderData · Rindermarkt 8 · D-8050 Freising · Telefon 08161/2877



für 464-664-6128



Es sollte wirklich kaum zu glauben sein, daß die ohnehin schon schnelle Schneider-Floppy noch zu beschleunigen ist.
Wer sich allerdings das ROM Listing der DD1 im Data Becker Floppy-Buch genauer ansieht, wird schnell erkennen, daß dieser Effekt mit relativ einfachen Mitteln zu erreichen ist.
Wie schon bekannt, beschäftigen sich bei Amstrad immer gleich mehrere Programmierer mit einem Programm. Genauso muß es auch beim AMSDOS gewesen sein. Die Grundwerte für Verzögerungszeiten bei Schreib- und Leseprozeduren wurden hier nämlich sehr willkürlich gewählt.

Sie können durch ein kurzes Assemblerlisting modifiziert werden. Ab Adresse C5D4 im ROM 7, dem Floppy-ROM, steht eine Tabelle in der die Parameter für die Laufzeiten festgelegt sind.

Diese Werte sind wie folgt:

adresse	wert	bemerkung
C5D4	0032	Hochlaufzeit fuer diskmotor
C5D6	00FA	Tickerlaufzeit fuer floppymotor
C5D8	AF	???????
C5D9	0F0C	Delayloop
C5DB	01	Head Unload Time
C5DC	03	Head Load Time

Die Werte in dieser Tabelle scheinen rein willkürlich gewählt zu sein, wenn man alle Werte durch Ausprobieren ein wenig manipuliert, so ergeben sich bei gleicher Datensicherheit wesentlich höhere Geschwindigkeiten bei jeder Floppyoperation.

Die nachfolgenden Werte ergeben eine Geschwindigkeitssteigerung von 25%. Dropouts oder Datenverluste verlassen hierbei auch nicht den gewohnten Rahmen, und auch die mechanische Beanspruchung des Laufwerks ist nicht zu hoch. Lediglich die Fehlermeldung DISK MISSING erscheint schneller als gewohnt. Durch Drücken der Taste R wird jedoch ein neuer Leseversuch vorgenommen, der das gewünschte Ergebnis bringt.

Folgende Werte sind also zu empfehlen:

C5D4	0023	hochlaufzeit Diskmotor
C5D6	00C8	Tickerlaufzeit Floppymotor
C5D8	01	????
C5D9	010A	Delayloop
C5DB	00	Head Unload Time
C5DC	03	Head Load Time

Natürlich kann man auch mit noch kleineren Werten arbeiten, aber dann geht das Ganze auf Kosten der Datensicherheit. Dem Experimentieren sind hierbei jedoch keine Grenzen gesetzt.
Stellt sich nur noch die Frage, wie wir die neue Tabelle dem DOS klarmachen. In ein ROM läßt sich ja nichts poken. Dazu machen wir einfach einen RST &18 zur Adresse C60D im Floppy-ROM.
Diese Adresse wird jedesmal beim Einschalten angesprungen und erfordert in HL die neue Tabelle. Also laden wir einfach HL mit der Tabelle, springen über eine ROM-CALL diese Adresse an und übergeben die neuen Werte. Den Rest erledigt das Floppy-ROM auf so komfortable Weise, daß durch unsere Routine nicht ein einziges Byte Speicherplatz verbraucht wird und tatsächlich bis zum Ausschalten der Floppy aktiv bleibt.
Die Vorteile für dieses Verfahren liegen auf der Hand. Kurzes Programm, keine Kollision mit bestehenden Programmen, volle Ausnutzung der Floppygeschwindigkeit.
Tippen Sie also den kleinen Basiclader ein und speichern Sie ihn ab, bevor Sie ihn starten. Der Lader löscht sich nach der Installation der neuen Parameter von selbst. Für MC-Interessierte ist das sehr kurze Assemblerlisting abgedruckt. Das Label ROMCAL spezifiziert hier die Adresse und ROM-Nummer, die angesprungen werden soll. Das Label NEWTAB enthält die neuen Laufwerksdaten.
Viel Spaß beim beschleunigten Arbeiten mit dem Schneider Laufwerk.

Volker Schick

```
10 FOR adr= &1000 TO &1012
20 READ byte$:byte=VAL("&"+byte$)
30 POKE adr,byte
40 NEXT adr
50 CALL &1000:CLS:PRINT"TURBO DOS 1.0 ON
":NEW
60 DATA 21,A,10,DF,7,10,C9,D,C6,7,23,0,C
8,0,1,1,A,0,3,0
```

```
BIG ASS         PASS 1

NO ERROR


1000  210A10              LD HL,NEWTAB
1003  DF0710              RST 18H,ROMCAL
1006  C9                  RET
1007  0DC607    ROMCAL:DB 0DH,C6H,07H
100A  2300C800NEWTAB:DB 23H,00H,C8H,00H
100E  01010A00            DB 1H,1H,AH,00

NO ERROR
```

DISCMON.BAS

für 464-664-6128



1. Allgemeines:
Nach Start mit RUN wird die Maschinenroutine "DISCMON.BIN" in den Bereich &6000-&608a geladen. Das Hauptmenue erscheint. Der Pufferbereich, in den die Sektoren mit der Länge 512 Byte geladen werden, liegt bei &6100-&62ff.

In Menues wird durch Tippen eines der invers dargestellten Buchstaben die gewünschte Funktion aufgerufen. Bei Zahleneingaben wird der zuletzt eingestellte Wert in Klammern dargestellt. Er kann übernommen werden, indem nur ENTER gedrückt wird.

2. Hauptmenue:
Das Programm ist modulartig aufgebaut, alle Vorgänge können in beliebiger Reihenfolge aufgerufen werden:

* Lesen von Diskette (l):
Nach Abfrage von Track und Sektor wird die erste Hälfte des Sektors, also 256 Byte, in HEXA und ASCII dargestellt. Das Lesemenue bietet folgende Möglichkeiten:

- "Leertaste" blättert weiter, zunächst in die zweite Hälfte des Sektors, dann in den nächsten Sektor, was einen Lesevorgang und Überschreiben des Puffers bedeutet.
- "z" blättert zurück.
- "a" verzweigt direkt in die Routine zum Ändern des Puffers, s. "Puffer ansehen und ändern".
- "d" druckt den Sektor aus, wie Hauptmenuepunkt.
- "m" kehrt ins Hauptmenue zurück.

* Lesen blockorientiert (b):
Da der Platz auf der Diskette in Blöcken von 1K, also zwei Sektoren, verwaltet wird, ist dies vor allem beim fileorientierten Arbeiten zweckmäßig. Die Blöcke, die ein File belegt, stehen im Directory in der Zeile unter dem Filenamen. Das Directory beginnt in Block 0.

* Schreiben auf Diskette (s):
Es werden wieder Sektor und Track abgefragt, was jeweils einfach mit ENTER beantwortet wird, wenn ein geänderter Sektor an die gleiche Stelle auf der Diskette zurückgeschrieben werden soll. Natürlich kann auch an einen anderen Ort oder auf eine andere Diskette kopiert werden.

* Puffer ansehen und ändern (a):
Der zuletzt eingestellte Puffer wird angezeigt wie beim Lesen, es erscheint auch das gleiche Menue. Im Gegensatz zum Lesen kann nur innerhalb des Sektors geblättert werden.

Das Ändern erfolgt folgendermaßen:
Es wird zunächst die Adresse angegeben, ab der geändert werden soll. Die Angabe erfolgt in HEXA, bezogen auf den Pufferanfang.
Anschließend wird die neue Zeichenkette eingegeben, wobei solange ASCII-Klartext einzugeben ist, solange nicht mit CTRL+x in HEXA-Eingabe umgeschaltet wurde. Bei HEXA-Eingabe werden je zwei Zeichen als ein Byte interpretiert. Mit CTRL+x kann auch wieder von HEXA in ASCII umgeschaltet werden, so daß beliebig gemischt werden kann.
Nach Abschluß mit ENTER wird die Zeichenkette ab der eingegebenen Adresse in den Puffer kopiert.

* Puffer ausdrucken (d):
Der Drucker wird auf Elite eingestellt, Ränder werden festgelegt und der amerikanische Zeichensatz wird gewählt. Die Steuerzeichen sind Epson-kompatibel. Änderungen je nach Geschmack sind leicht möglich (Zeile 1710).

* Puffer festlegen (p):
Die voreingestellte Pufferlage (&6100-&62ff) kann auf höhere Adressen verändert werden. Hierdurch können mehrere Sektoren in den Arbeitsspeicher übernommen werden.

* CAT (c): Normale Directory-Ausgabe
* Ende (e): Programm verlassen

3. Anwendungsbeispiele:
Mit der Möglichkeit, den Disketteninhalt unabhängig von den Einschränkungen des Basic's zu verändern, ergibt sich ein weites Anwendungs- und Experimentierfeld. Beim Ausprobieren sollte man aber eine Kopie der Diskette gemacht haben.

- Wiederherstellen von gelöschten Files:
Beim Löschen eines Files wird zunächst nur das erste Byte des File-Eintrages im Directory (ab Track 2, Sector 1) auf &E5 gesetzt. Solange nicht durch erneutes Erzeugen eines Files der frei gewordene Platz auf der Diskette beansprucht wird, kann durch Eintrag einer User-Nr. zwischen &00 und &0f anstelle von &E5 das File wieder hergestellt werden.

- Setzen von read/only bzw. read/write:
Durch Setzen des höchsten Bits im ersten Zeichen der Extension des Filenamens wird das File vor Löschen und Überschreiben geschützt. Das entspricht einer Addition von &80, also wird z.B. aus &42 in der Extension "BAS" ein &C2.

- Setzen von Systemeigenschaft:
Durch entsprechendes Verändern des zweiten Zeichens in der Extension wird das File nicht mehr unter CAT oder DIR angezeigt. Das Rücksetzen erfolgt jeweils entsprechend.

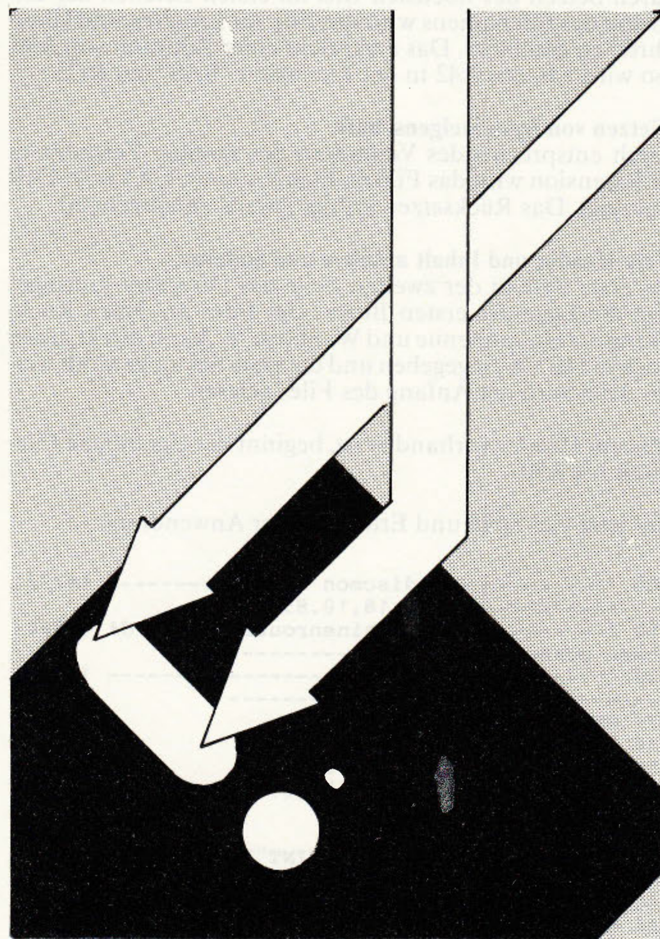
- File-Header und Inhalt ansehen und ändern:
Das erste Byte in der zweiten Zeile des Directory-Eintrages zeigt die Lage des ersten Blocks des Files an. Nach Rücksprung ins Hauptmenue und Wahl von "b" kann direkt dieser Block in HEXA eingegeben und der erste Sektor gewählt werden. Jetzt wird der Anfang des Files gelesen.

Falls ein Header vorhanden ist, beginnt der eigentliche File-Inhalt bei &80.

Und jetzt viel Spaß und Erfolg bei der Anwendung:

```
100 '------------ pdiscmon --------------- [4732]
-------------------- 18.10.85 ---
110 '------------- Maschinenroutine fuer di [5860]
scmon poken ---------------------
120 '------------------------------------- [3447]
---------------------------------
130 MEMORY &5FFF                           [102]
140 RESTORE 210 : sum=0                    [752]
150 FOR adr=&6000 TO &6089                 [1262]
160     READ i$ : i=VAL("&"+i$) : sum=sum+i [2936]
 : POKE adr,i
170 NEXT adr                               [453]
180 IF sum <> 13540 THEN PRINT"Fehler in C [3300]
heck-Summe!"CHR$(7) : STOP
190 SAVE"discmon.bin",b,&6000,&8B          [2000]
200 END                                    [110]
210 DATA 2a,7f,60,7d,fe,85,20,0c,11,5e,60, [2797]
cd,3d,60,cd,06
220 DATA bb,fe,6a,c0,21,7f,60,cd,d4,bc,38, [2877]
07,11,47,60,cd
230 DATA 3d,60,c9,22,80,60,79,32,82,60,1e, [2363]
00,16,02,0e,41
240 DATA 21,00,61,df,80,60,21,84,00,22,7f, [4050]
60,c9,1a,fe,24
250 DATA c8,cd,5a,bb,13,18,f6,63,6f,6d,6d, [2874]
61,6e,64,20,6e
260 DATA 69,63,68,74,20,67,65,66,75,6e,64, [3596]
65,6e,24,73,65
270 DATA 63,74,6f,72,20,61,75,66,20,64,69, [1891]
73,63,20,73,63
280 DATA 68,72,65,69,62,65,6e,3f,20,28,6a, [2939]
2f,6e,29,24,84
290 DATA 00,00,00,cd,06,bb,32,ff,60,c9     [2550]
```

```
100 '--------- discmon -------------------- [4306]
------------- 18.10.85 -----------
110 '---------- Direktes Lesen und Schreibe [5714]
n von Disketten-Sectoren -------------
120 '------------------------------------- [3711]
-----------------------------------------
130 MEMORY &5FFF : DEFINT a-z : MODE 2      [1822]
140 LOAD"discmon.bin",&6000                 [5092]
             'Maschinenroutine laden
150 buffer=&6100 : a$=STRING$(16," ") : u$ [4018]
=CHR$(24) : lf$=CHR$(13)+CHR$(10)
160 k=0 : sector=65 : sbloc=1 : x$="&0000" [4149]
 : del$=CHR$(8)+CHR$(18) : b$=CHR$(7)
170 lb$=CHR$(11)+CHR$(18)                   [1164]
180 l$="      0   1   2   3   4   5   6   7   8 [3383]
9  A  B  C  D  E  F  0123456789ABCDEF"
190 in=32 : taste=&6083 : POKE taste+4,@in [4986]
 MOD 256 : POKE taste+5,@in\256
200 GOSUB 1230                              [877]
210 '                                       [117]
220 '-------------- Hauptmenue ------------ [3772]
------------------------------
230 CLS                                     [91]
240 PRINT "*********************** Disc - [5865]
 Monitor *****************************
****" : PRINT lf$
250 PRINT "                   Lesen von Dis [3567]
c:          "u$" l "u$lf$
260 PRINT "                   Lesen Block-o [4293]
```



```
rientiert: "u$" b "u$lf$
270 PRINT "                   Schreiben auf [5120]
 Disc:      "u$" s "u$lf$
280 PRINT "                   Puffer ansehe [4342]
n+aendern:  "u$" a "u$lf$
290 PRINT "                   Puffer ausdru [3359]
cken:       "u$" d "u$lf$
300 PRINT "                   Puffer festle [4140]
gen:        "u$" p "u$
 "      (&"HEX$(buffer)"-&"HEX$(buffer+&1
FF)")"lf$
310 PRINT "                   CAT:          [2726]
            "u$" c "u$lf$
320 PRINT "                   Ende:         [3088]
            "u$" e "u$lf$lf$
330 CALL taste                              [748]
340 IF in=108 THEN GOSUB 1020 : GOSUB 1300 [2387]
 : GOTO 230
350 IF in= 98 THEN GOSUB 1100 : GOSUB 1300 [2933]
 : GOTO 230
360 IF in=115 THEN GOSUB 1020 : POKE &607F [3473]
,&85 : CALL &6000 : GOTO 230
370 IF in= 97 THEN GOSUB 1400 : GOTO 230   [1773]
380 IF in=100 THEN GOSUB 1700 : GOTO 230   [2062]
390 IF in=112 THEN GOSUB 1200 : GOTO 230   [1361]
400 IF in=99 THEN CLS : CAT : PRINT "Leert [4609]
aste druecken": CALL taste : GOTO 230
410 IF in=101 THEN CLS : END               [999]
420 PRINT CHR$(7); : GOTO 330              [1217]
1000 '                                     [117]
1010 '--------------- Eingabe von track,   [5091]
sector ------------------------------
1020 PRINT "track"; : x=track    : GOSUB  [3429]
1900 : track=x
1030 IF track<0 OR track>39 THEN b$l [4681]
b$"nur 0-39 moeglich!   "; : GOTO 1020
1040 PRINT "sector"; : x=sector-64: GOSUB [3403]
1900 : sector=x+64
1050 IF sector<65 OR sector>73 THEN PRINT [3457]
b$lb$"nur 1-9 moeglich!   ";:GOTO 1040
1060 drive=0 : POKE &602B,drive           [2945]
1070 POKE &602D,track : POKE &602F,sector [2719]
1080 RETURN                               [555]
1100 '--------------- Eingabe von Block,   [4513]
sector -----------------------------
1110 IF track>1 THEN x=(track-2)*9+sector- [4967]
65 : block=x\2 : sbloc=x MOD 2 +1
1120 PRINT "Block       "; : x=block : GOSUB [3217]
 2000 : block=x
1130 PRINT "sector 1/2"; : x=sbloc  GOSUB [4634]
 1900 : sbloc=x
1140 x=block*2+17+sbloc : track=x\9 : sect [4832]
or=x MOD 9 +65
1150 GOSUB 1070                           [829]
1160 RETURN                               [555]
1200 '-------------------- Puffer defini [5399]
eren -----------------------------------
1210 INPUT"Neuer Buffer-Anfang? (in Hex)"; [3596]
buf$
1220 buffer=VAL("&"+buf$)                 [2227]
1230 IF buffer<6100 THEN PRINT "Adresse zu [4439]
 klein!"CHR$(7) : GOTO 1210
1240 POKE &6031,buffer MOD 256 : POKE &603 [3865]
2,buffer\256
1250 RETURN                               [555]
1300 '---------------------- Lesen von D [3776]
isc ---------------------------------
1310 CALL &6000          'sector in buffer le [2105]
sen
1320 GOSUB 1400                           [899]
1330 IF in= 32 THEN 1360                  [1227]
1340 IF in=122 THEN 1380                  [583]
1350 RETURN                               [555]
1360 IF sector<73 THEN sector=sector+1 ELS [6695]
E IF track<39 THEN track=track+1 :
   sector=65 ELSE PRINT b$;
1370 GOTO 1390                            [421]
1380 IF sector>65 THEN sector=sector-1 ELS [7061]
E IF track>0 THEN track=track-1
   sector=73 : ELSE PRINT b$;
1390 GOSUB 1070 : GOTO 1310               [1462]
1400 '----------------------- Puffer anz [4895]
eigen -------------------------------
1410 anfg=buffer : i2=1                   [1642]
1420 IF k=0 THEN CLS                      [1587]
1430 PRINT#k,"track";track;"  sector"secto [3314]
r-64;
1440 IF k=0 THEN PRINT "  part";i2;lf$lf$l [2747]
$lf$ ELSE PRINT#k,lf$lf$l$lf$
1450 FOR i=anfg TO anfg+&FF STEP &10      [2070]
1460   MID$(a$,1)=STRING$(16,".")         [1037]
1470   PRINT#k," ";HEX$(i-buffer,4);" ";  [2696]
1480     FOR l=0 TO &F                    [1126]
1490     a=PEEK(i+l) : PRINT#k,HEX$(a,2);" [4493]
 ";: IF a>31 AND a<127 THEN
    MID$(a$,l+1)=CHR$(a)
1500     NEXT l                           [464]
1510 PRINT#k," "a$                        [1196]
1520 NEXT i                               [471]
1530 IF k=8 THEN RETURN                   [983]
1540 PRINT lf$"weiter:"u$"Leertaste"u$"   [9725]
```

```
zurueck:"u$" z "u$"   aendern:"u$" a "
      u$"  drucken:"u$" d "u$"   Menue:"
u$" m "u$
1550 CALL taste                          [748]
1560 IF in=32 AND i2=1 THEN i2=2 : anfg=bu [4048]
ffer+&100 : GOTO 1420
1570 IF in=122 AND i2=2 THEN i2=1 : anfg=b [3165]
uffer : GOTO 1420
1580 IF in=109 OR in=32 OR in=122 THEN RET [3048]
URN
1590 IF in=97 THEN GOSUB 1760 : GOTO 1420  [3109]
1600 IF in=100 THEN GOSUB 1700            [1644]
1610 PRINT CHR$(7); : GOTO 1550           [1558]
1620 '                                    [117]
1700 '--------------- Puffer ausdrucken -- [3671]
------------------------------------
1710 PRINT #8,CHR$(27)"B"CHR$(2)CHR$(27)"R [9565]
"CHR$(4)CHR$(27)"N"CHR$(4)CHR$(27)"M"CHR$(
8)CHR$(27)"7"CHR$(0);     'Drucker: Elite,
 Raender, amerikan. Zeichensatz
1720 k=8 : GOSUB 1400                     [1159]
1730 anfg=buffer+&100 : GOSUB 1450        [2002]
1740 k=0                                  [570]
1750 RETURN                               [555]
1760 '--------------- Puffer veraendern -- [4617]
------------------------------------
1770 PRINT lb$"ab welchem byte im Puffer a [6375]
endern, hexa   ";: x=start :
 GOSUB 2000 : start=x
1780 PRINT lb$"neuer Inhalt ab &"HEX$(star [8001]
t)"? (Umschalten in hexa und zurueck mit C
TRL+x)"lf$ : LINE INPUT strp$
1790 hex=0 : ip=0                         [1364]
1800 FOR istr=1 TO LEN(strp$)             [1896]
1810    ap=ASC(MID$(strp$,istr,1)) : adr=bu [4516]
ffer+start+ip
1820    IF ap=24 THEN IF hex=0 THEN hex=-1 [4559]
: istr=istr+1 ELSE hex=0 : GOTO 1880
1830    IF hex THEN 1860                  [1644]
1840    POKE adr,ap                       [1174]
                'ascii
1850    GOTO 1870                         [349]
1860    POKE adr,VAL("&"+MID$(strp$,istr,2) [3041]
) : istr=istr+1  'hexa
1870 ip=ip+1                              [1218]
1880 NEXT istr                            [864]
1890 RETURN                               [555]
1900 '------------- Eingabe von Dec-Zahlen [4169]
------------------------------------
1910 PRINT " (";x;") ? ";                 [1238]
1920 z=1 : GOSUB 1940 : IF in=13 THEN PRIN [2715]
T : RETURN ELSE x=0
1930 x=x*z+in-48 : z=z*10 : GOSUB 1940 : I [3671]
F in=13 THEN PRINT : RETURN ELSE 1930
1940 CALL taste                           [748]
1950 IF in=13 THEN RETURN ELSE IF in=127 T [6121]
HEN 1980 ELSE IF in<48 OR in>57 THEN
    PRINT CHR$(7);: GOTO 1940
1960 PRINT CHR$(in);                      [878]
1970 RETURN                               [555]
1980 IF z>1 THEN PRINT CHR$(8)CHR$(18); : [3920]
x=x\10 : z=z\10 ELSE x=0
1990 GOTO 1940                            [343]
2000 '------------ Eingabe von Hexa-Zahlen [4942]
------------------------------------
2010 PRINT " ( ";HEX$(x)" ) ? ";          [1752]
2020 z=2 : GOSUB 2040 : IF in=13 THEN PRIN [5395]
T : RETURN ELSE MID$(x$,2)="  "
2030 MID$(x$,z)=CHR$(in) : z=z+1 : GOSUB 2 [3682]
040 : IF in=13 THEN 2080 ELSE 2030
2040 CALL taste                           [748]
2050 IF in=13 THEN RETURN ELSE IF in=127 T [9836]
HEN 2100 ELSE IF in<48 OR in>70 AND
    in<96 OR in>57 AND in<65 OR in>102 THE
N PRINT CHR$(7); : GOTO 2040
2060 PRINT CHR$(in);                      [878]
2070 RETURN                               [555]
2080 x=VAL(UPPER$(x$)) : PRINT            [1845]
2090 RETURN                               [555]
2100 IF z>2 THEN PRINT del$; : z=z-1 : MID [3175]
$(x$,z)=" "
2110 GOTO 2040                            [355]
```

DiscTape als RSX

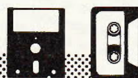
Der RSX-Befehl "|DISCTAPE" ist dem umständlichen CP/M-Befehl "CSAVE" nachempfunden. Im CP/M muß das CSAVE-Programm und das zu kopierende Programm auf einer Diskette sein.

Bei diesem RSX wird erst der Name des Programms auf Diskette übergeben, dann der Name, mit dem es auf Kassette gespeichert werden soll. Zum Schluß wird noch die Geschwindigkeit übergeben. Sie muß eine Zahl zwischen 130 und 480 sein. Bei 167 hat man eine Geschwindigkeit von 2000 Baud (speed write 1), bei 333 eine von 1000 Baud (speed write 0).
Beispiel:
a$="test.bin" : b$="testprogramm"
|disctape,@a$,@b$,167

Das Programm "test.bin" auf Diskette wird auf Kassette unter dem Namen "testprogramm" mit 2000 Baud abgespeichert. Headerlose Programme lassen sich leider nicht übertragen.

Guido Trier

für 464-664-6128

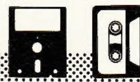


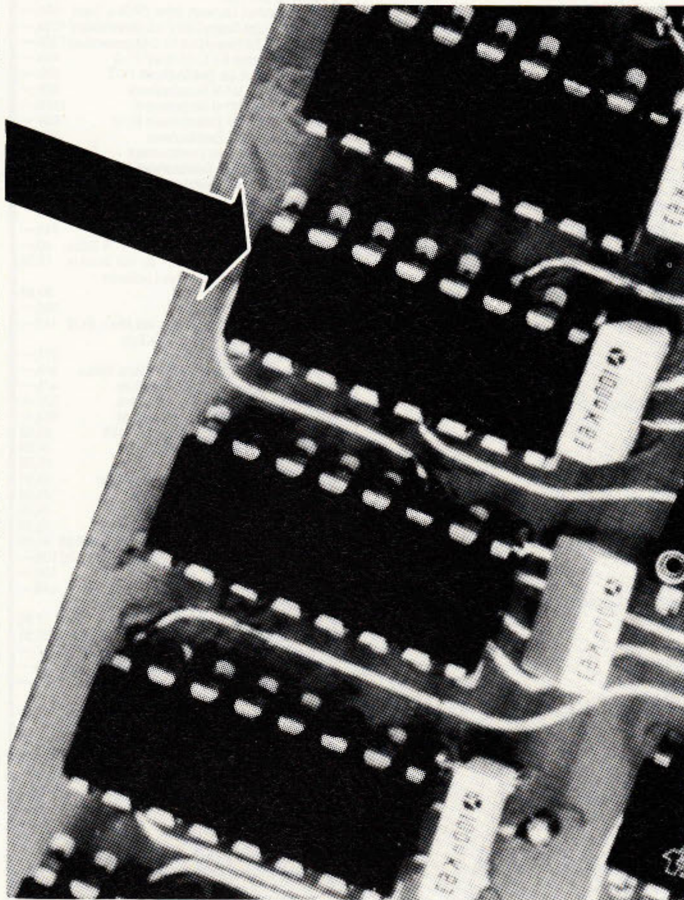
```
10 ' ****************************        [1164]
20 ' *     RSX : "DISCTAPE"     *        [1203]
30 ' *    (c) by Guido Trier    *        [805]
40 ' *        1986/Bonn         *        [497]
50 ' ****************************        [1164]
60 MEMORY &A0FF                                [446]
70 add=&A100 : RESTORE 150 : summe=0           [1523]
80 READ wert$ : IF wert$="999" THEN 100        [2326]
90 wert=VAL(wert$) : summe=summe+wert : PO     [5210]
KE add,wert : add=add+1 : GOTO 80
100 CLS : IF summe<>21104 THEN 110 ELSE 13     [2545]
0
110 PRINT "Error in Checksum !!!"              [2924]
120 END                                        [110]
130 PRINT "Alles OK !!!"                       [1353]
140 CALL &A100 : END                           [737]
150 DATA &1,&10,&A1,&21,&A,&A1,&CD,&D1,&BC     [3800]
,&C9,&0,&0,&0,&0,&0,&0
160 DATA &15,&A1,&C3,&1E,&A1,&44,&49,&53,&     [2582]
43,&54,&41,&50,&C5,&0,&DD,&66
170 DATA &1,&DD,&6E,&0,&22,&0,&AC,&DD,&66,     [2907]
&3,&DD,&6E,&2,&7E,&32,&25
180 DATA &AC,&23,&5E,&23,&56,&ED,&53,&20,&     [1947]
AC,&DD,&66,&5,&DD,&6E,&4,&7E
190 DATA &32,&15,&AC,&23,&5E,&23,&56,&ED,&     [3145]
53,&10,&AC,&21,&B7,&A1,&CD,&D4
200 DATA &BC,&AF,&CD,&1B,&0,&2A,&10,&AC,&3     [2626]
A,&15,&AC,&47,&11,&0,&40,&CD
210 DATA &77,&BC,&ED,&53,&30,&AC,&ED,&43,&     [2989]
40,&AC,&32,&60,&AC,&2A,&30,&AC
220 DATA &CD,&83,&BC,&22,&50,&AC,&CD,&7A,&     [4212]
BC,&21,&BB,&A1,&CD,&D4,&BC,&AF
230 DATA &CD,&1B,&0,&2A,&0,&AC,&3E,&32,&CD     [2148]
,&68,&BC,&3A,&25,&AC,&47,&2A
240 DATA &20,&AC,&11,&0,&40,&CD,&8C,&BC,&2     [3180]
A,&30,&AC,&ED,&5B,&40,&AC,&ED
250 DATA &4B,&50,&AC,&3A,&60,&AC,&CD,&98,&     [3839]
BC,&CD,&8F,&BC,&21,&B7,&A1,&CD
260 DATA &D4,&BC,&AF,&CD,&1B,&0,&C9,&44,&4     [4022]
9,&53,&C3,&54,&41,&50,&C5,999
```

```
  1 ;******************************
  2 ;*     RSX : "DISCTAPE"        *
  3 ;*    (c) by Guido Trier       *
  4 ;*        1986/Bonn            *
  5 ;*     Assembler-Listing       *
  6 ;******************************
  7 ;
  8 ;
  9 ;
 10          org   #a100                ;Programmanfang auf #a100
 20 buffer:  equ   #4000                ;Buffer auf #4000
 30          ld    bc,comand            ;Adresse des Kommando in BC
 40          ld    hl,kern              ;Adresse des Kern in HL
 50          call  #bcd1                ;Einbinden des RSX-Befehl
 60          ret
 70 kern:    defs  #0006                ;Vom System benutzter Kern fuer RSX
 80 comand:  defw  tabell               ;Adresse der Kommandotabelle
 90          jp    routin               ;Sprungtabelle
100 tabell:  defm  "DISCTAP"            ;Name des RSX
110          defb  #c5                  ;letzter Buchstabe+#80
120          defb  0                    ;Kennzeichen fuer Ende der Tabelle
125 ;======= PARAMETER-UEBERGABE =======
130 routin:  ld    h,(ix+#01)
140          ld    l,(ix+#00)
160          ld    (#ac00),hl           ;Geschwindigkeit nach #ac00
170          ld    h,(ix+#03)
180          ld    l,(ix+#02)
181          ld    a,(hl)
182          ld    (#ac25),a            ;Laenge des Kassettennamen nach #ac25
185          inc   hl
190          ld    e,(hl)
200          inc   hl
210          ld    d,(hl)
220          ld    (#ac20),de           ;Adresse des Kassettennamen nach #ac20
230          ld    h,(ix+#05)
240          ld    l,(ix+#04)
241          ld    a,(hl)
242          ld    (#ac15),a            ;Laenge des Diskettennamen nach #ac15
245          inc   hl
250          ld    e,(hl)
260          inc   hl
270          ld    d,(hl)
280          ld    (#ac10),de           ;Adresse des Diskettennamen nach #ac10
285 ;======= UMSCHALTEN AUF DISC =======
290          ld    hl,disc
300          call  #bcd4                ;suchen nach Befehl  DISC
310          xor   a                    ;kein Parameter ist uebergeben
320          call  #001b                ;ausfuehren des Befehls  DISC
325 ;======= HEADER LESEN UND PROGRAMM LADEN =======
330          ld    hl,(#ac10)           ;Adresse des Diskettennamen in HL
331          ld    a,(#ac15)
340          ld    b,a                  ;Laenge des Diskettenname in B
350          ld    de,buffer            ;Adresse des Buffers in DE
360          call  #bc77                ;DISC IN OPEN
370          ld    (#ac30),de           ;Anfangsadresse des Programm nach #ac30
380          ld    (#ac40),bc           ;Laenge des Programms nach #ac40
390          ld    (#ac60),a            ;Art des Programms nach #ac60
400          ld    hl,(#ac30)           ;Anfangsadresse des Programm in HL
410          call  #bc83                ;Programm laden
420          ld    (#ac50),hl           ;Einsprungadresse nach #ac50
430          call  #bc7a                ;DISC IN CLOSE
435 ;======= UMSCHALTEN AUF TAPE =======
440          ld    hl,tape
450          call  #bcd4                ;suchen nach Befehl  TAPE
460          xor   a                    ;kein Parameter ist uebergeben
470          call  #001b                ;ausfuehren des Befehls  TAPE
475 ;======= GESCHWINDIGKEIT SETZEN =======
480          ld    hl,(#ac00)           ;Geschwindigkeit in HL
490          ld    a,50                 ;Vorprueflaenge in A
500          call  #bc68                ;Geschwindigkeit  setzen
501 ;======= PROGRAMM SPEICHERN =======
505          ld    a,(#ac25)
510          ld    b,a                  ;Laenge des Kassettenname in B
520          ld    hl,(#ac20)           ;Adresse des Kasettenname in HL
530          ld    de,buffer            ;Adresse des Buffers in DE
540          call  #bc8c                ;CAS OUT OPEN
550          ld    hl,(#ac30)           ;Anfangsadresse des Programms in HL
560          ld    de,(#ac40)           ;Laenge des Programms in DE
570          ld    bc,(#ac50)           ;Einsprungadresse des Programms in BC
580          ld    a,(#ac60)            ;Programmart in a
590          call  #bc98                ;Programm speichern
600          call  #bc8f                ;CAS OUT CLOSE
605 ;======= UMSCHALTEN AUF DISC =======
610          ld    hl,disc
620          call  #bcd4                ;suchen nach Befehl  DISC
630          xor   a                    ;kein Parameter ist uebergeben
640          call  #001b                ;ausfuehren des Befehls  DISC
650          ret
660 disc:    defm  "DIS"                ;Name fuer DISC
670          defb  #c3                  ;letzter Buchstabe+#80
680 tape:    defm  "TAP"                ;Name fuer TAPE
690          defb  #c5                  ;letzter Buchstabe+#80
```

MC-Tracer

für 464-664-6128





MC-TRACER ist ein Hilfsprogramm für alle CPC-Besitzer, die mehr über das ROM Ihres Computers wissen möchten. Es ermöglicht die Disassemblierung und Einzelschrittverarbeitung beliebiger ROM- und RAM-Bereiche.

Nach dem Start des Programmes wird man nach dem ROM-Select-Byte gefragt, mit dieser Zahl zwischen 0 und 255 bestimmt man, ob an den Adressen 0 bis 16383 (Bank 0) und 49152 bis 65535 (Bank 3) der ROM- oder RAM-Speicher gelesen werden kann.

Durch die Eingabe von 'Z' gelangt man zum Tracer, hier muß zuerst bestimmt werden, ob ein CALL-Befehl weiterverfolgt und ob die Z80-Macrobefehle, wie LDIR und LDDR, in Einzelbefehle aufgelöst abgearbeitet werden sollen. Nach der Eingabe der gewünschten Speicheradresse, wird der erste Maschinencodebefehl disassembliert, die aktuellen Registerwerte des Z80 und die fünf obersten Einträge auf dem Stapel angezeigt. Danach kann man entscheiden, ob der Programmablauf fortgesetzt oder abgebrochen werden soll. Mit 'ENTER' erreicht man die Fortsetzung, jede andere Taste führt zum Abbruch der Trace-Funktion. Mit 'D' kann man beliebige Speicherbereiche disassemblieren, das erzeugte Assemblerlisting kann man wahlweise auf dem Drucker oder auf dem Bildschirm ausgeben lassen.

Es werden immer 23 Zeilen nacheinander ausgegeben, mit 'ENTER' kann man die Ausgabe dann fortsetzen, eine andere Taste bricht die Ausgabe ab.

Nach der Eingabe von 'S' kann man einen neuen ROM/RAM-Status wählen und mit Hilfe von 'A' können die Registerwerte des Z80 Mikroprozessors geändert werden. Die Register können einzeln oder paarweise geändert werden. Die möglichen Bezeichnungen für die Einzelregister sind: A, B, C, D, E, H, L, F, I, R, A', B', C', D', E', H', L' und F'.

Die möglichen Bezeichnungen für die Doppelregister sind: AF, BC, DE, HL, IR, IX, IY, SP, AF', BC', DE' und HL'. So kann man z.B. den Stapelzeiger mit der Eingabe von 'SP' und '&8B00' wieder auf seinen ursprünglichen Wert zurücksetzen.

Anweisung zur Erzeugung einer schlüsselfertigen Version anhand der Programmlistings:
Zuerst das Programm TRACER abtippen und abspeichern, danach die Programme OPCODE und MCPROG in dieser Reihenfolge abtippen und ablaufen lassen und den von diesen beiden Programmen erzeugten Code direkt hinter TRACER abspeichern, anschließend muß dann noch das Basic-Programm BASPROG dahinter abgespeichert werden.

Thomas Fippl

Programm TRACER

```
100 SYMBOL AFTER 256                          [14
08]
110 POKE &AE7D,&FF                            [475]
120 POKE &AE7E,&7F                            [816]
130 POKE &B296,&FF                            [529]
140 POKE &B297,&7F                            [407]
150 MEMORY &3FFF                              [758]
160 LOAD"!code",&8000                         [659]
170 CALL &8513                                [516]
180 RUN"!basprog                              [1111]
```

Programm OPCODE

```
100 l=&8000:z=250:MEMORY &7FFF                [2142]
110 FOR n=1 TO 282:READ b$                    [1399]
120 s=VAL(b$):IF s GOTO 220                   [2181]
130 FOR m=1 TO LEN(b$)-1                      [1907]
140 a=ASC(MID$(b$,m))                         [991]
150 p=(p XOR a)+a                             [1178]
160 IF a=59 THEN a=44                         [1114]
170 POKE l,a:l=l+1:NEXT                       [973]
180 b=ASC(MID$(b$,m)):a=b+128                 [1986]
190 p=(p XOR a)+b                             [1185]
200 POKE l,a:l=l+1                            [684]
210 NEXT:END                                  [735]
220 IF s<>p THEN PRINT"Error in";z:STOP       [2429]
230 p=0:z=z+1:GOTO 210                        [1070]
240 '                                         [117]
250 DATA B,C,D,E,H,L,(HL),A,HALT,ADD,ADC,S    [3642]
UB,SBC,AND,2990
251 DATA XOR,OR,CP,RLC,RRC,RL,RR,SLA,SRA,S    [3003]
LI,SRL,BIT,1974
252 DATA RES,SET,NOP,LD BC;m,LD (BC);A,INC    [2552]
BC,2726
253 DATA INC B,DEC B,LD B;n,RLCA,EX AF;AF'    [1695]
,ADD HL;BC,2142
254 DATA LD A;(BC),DEC BC,INC C,DEC C,LD C    [2360]
;n,3014
255 DATA RRCA,DJNZ e,LD DE;m,LD (DE);A,INC    [3816]
DE,2034
256 DATA INC D,DEC D,LD D;n,RLA,JR e,ADD H    [1756]
L;DE,2516
257 DATA LD A;(DE),DEC DE,INC E,DEC E,LD E    [2478]
;n,3170
258 DATA RRA,JR NZ;e,LD HL;m,LD (m);HL,INC    [2186]
HL,2468
259 DATA INC H,DEC H,LD H;n,DAA,JR Z;e,ADD    [2978]
HL;HL,2590
260 DATA LD HL;(m),DEC HL,INC L,DEC L,LD L    [2818]
;n,2180
261 DATA CPL,JR NC;e,LD SP;m,LD (m);A,INC     [2111]
SP,2144
262 DATA INC (HL),DEC (HL),LD (HL);n,SCF,J    [2476]
R C;e,3054
263 DATA ADD HL;SP,LD A;(m),DEC SP,INC A,D    [697]
EC A,1912
264 DATA LD A;n,CCF,RET NZ,POP BC,JP NZ;m,    [2576]
JP m,2244
265 DATA CALL NZ;m,PUSH BC,ADD n,RST 0,RET    [1496]
Z,3292
266 DATA RET,JP Z;m,>,CALL Z;m,CALL m,ADC     [2768]
n,RST 1 /DW m,2830
267 DATA RET NC,POP DE,JP NC;m,OUT (n);A,C    [2824]
ALL NC;m,3510
```

```
268 DATA PUSH DE,SUB n,RST 2 /DW m,RET C,E [2650]
XX,1884
269 DATA JP C;m,IN A;(n),CALL C;m,>,SBC n, [3589]
RST 3 /DW m,2354
270 DATA RET PO,POP HL,JP PO;m,EX (SP);HL, [2256]
CALL PO;m,3018
271 DATA PUSH HL,AND n,RST 4,RET PE,JP HL, [2494]
JP PE;m,926
272 DATA EX DE;HL,CALL PE;m,>,XOR n,RST 5  [2076]
/DW m,2080
273 DATA RET P,POP AF,JP P;m,DI,CALL P;m,P [3097]
USH AF,3254
274 DATA OR n,RST 6,RET M,LD SP;HL,JP M;m, [3257]
EI,3138
275 DATA CALL M;m,>,CP n,RST 7,IN B;(C),OU [2970]
T (C);B,2658
276 DATA SBC HL;BC,LD (m);BC,NEG,RETN,IM 0 [2250]
,LD I;A,2032
277 DATA IN C;(C),OUT (C);C,ADC HL;BC,LD B [2534]
C;(m),2470
278 DATA RETI,?,?,LD R;A,IN D;(C),OUT (C); [2411]
D,SBC HL;DE,2318
279 DATA LD (m);DE,?,?,IM 1,LD A;I,IN E;(C [2498]
),OUT (C);E,2774
280 DATA ADC HL;DE,LD DE;(m),?,?,IM 2,LD A [2286]
;R,2036
281 DATA IN H;(C),OUT (C);H,SBC HL;HL,LD ( [3107]
m);HL,1938
282 DATA ?,?,?,RRD,IN L;(C),OUT (C);HL,ADC [3308]
 HL;HL,1154
283 DATA LD HL;(m),?,?,?,RLD,?,?,SBC HL;SP [3405]
,LD (m);SP,2796
284 DATA ?,?,?,?,IN A;(C),OUT (C);A,ADC HL [3038]
;SP,1898
285 DATA LD SP;(m),LDI,CPI,INI,OUTI,?,?,?, [2806]
?,LDD,CPD,1150
286 DATA IND,OUTD,?,?,?,?,LDIR,CPIR,INIR,O [2721]
TIR,?,?,?,3402
287 DATA ?,LDDR,CPDR,INDR,OTDR,420         [3564]
```

Programm MCPROG

```
100 m=&84E8:FOR z=200 TO 274               [1539]
110 READ b$,s:p=0                          [761]
120 FOR l=1 TO LEN(b$)STEP 2               [1880]
130 b=VAL("&"+MID$(b$,l,2))                [1432]
140 POKE m,b:p=(p XOR b)+b                 [1841]
150 m=m+1:NEXT                             [1542]
160 IF s<>p THEN PRINT"error in";z:STOP    [2825]
170 NEXT                                   [350]
180 SAVE"code",b,&8000,&990                [1475]
190 '                                      [117]
200 DATA 00000000000000000000000000000000, [2053]
0
201 DATA 000000000000000008b0000000000000, [2311]
278
202 DATA 000000000000000000000012085211c, [1899]
430
203 DATA 85c3d1bc00000003185c34885c35285, [2252]
1894
204 DATA c35785c35c85c361854449d35553d253, [2535]
2686
205 DATA 45d45045cb5453d4000000000000000, [2189]
1452
206 DATA 21668522428 5df4285c9219b8718f421, [1666]
700
207 DATA f28718ef21fd8718ea212b8818e5fe02, [3175]
2912
208 DATA c0af3241852f324585dd4603dd4e02c5, [3514]
2204
209 DATA d5dde1cd5d87feddcc5887fefdcc5487, [2376]
3412
210 DATA fecbca3f86feedca77867ba7280d3dca, [2716]
2372
211 DATA 9b863dcac08621cd8118032144807ae6, [2253]
2594
212 DATA 3f4728077e238730fb10f91104857ee6, [2442]
1686
213 DATA 7f1213fe202810fe3f20061213121318, [3039]
1002
214 DATA 6abe2328e91864237e23f5e67f123a41, [2936]
1846
215 DATA 85a7283547eb7efe28200d1ae67ffe48, [1885]
1718
216 DATA 2005cdcc8618117efe4820181ae67ffe, [3189]
2160
217 DATA 4c200b3649237023ebf17e23182a7023, [2101]
1588
218 DATA 1a7718f3fe4c28f6ebe51afe652005cd, [2622]
2130
219 DATA 00871812fe6e2005cd16871809fe6d20, [2089]
1344
220 DATA 04cd1e871b13e1f187309de10e047b91, [1818]
1910
221 DATA 772371233685c93a4185a7280bdd23cd, [2255]
1178
222 DATA 5d87dd2bdd2b1803cd5d877b110485a7, [3070]
1692
223 DATA 200a21258078cd7e87eb1854c53d213b, [3510]
1670
224 DATA 80cd7e87f1c63012133e2c121318eacd, [3057]
2344
225 DATA 5d87d6403817fe3c3006215383c3a785, [2763]
1298
226 DATA d6603809fe1c300521a28418f021ae83, [2851]
1990
227 DATA c3b3857afe762005210b8018f3210485, [3112]
1434
228 DATA 364c2336442336202378cdd186362c23, [2170]
572
229 DATA 79cdd1865dc33386210f8011048578cd, [2165]
1710
230 DATA 7e871895c51313181ac51100804f3a41, [2706]
1706
231 DATA 8547a7792007ebcd8687ebc1c9fe0620, [2387]
2210
232 DATA f5362823364923702336 2b23ebe5cd16, [2737]
1556
233 DATA 87e1eb362923c1c9d5cd5d87dde52145, [1511]
1982
234 DATA 85cbf6e116005f87300115191816d5cd, [2041]
1808
235 DATA 5d876faf1810d5214585cbfecd5d876f, [2206]
2102
236 DATA cd5d87673e01d14f3e2612130d7ccc3a, [3757]
1366
237 DATA 877d4fe6f01f1f1f1fcd478779e60ffe, [3737]
2104
238 DATA 0a3804c6371802c6301213c93e591802, [2096]
952
239 DATA 3e58324185dd7e00dd23e521458534e1, [2312]
1768
240 DATA 57e6074f7a1f1f1fe607477a1e0017cb, [2586]
996
241 DATA 1317cb137ac9cd86873e201213c9a728, [3048]
1752
242 DATA 08477e238730fb10f97ee67f12be2313, [1864]
2020
243 DATA 28f7c93dc0f3ed73e884ed53c58731ea, [2197]
2762
244 DATA 84c179ed4778d612ed4ff108f1c1d1e1, [2672]
3214
245 DATA d9c1d1e1dde1fde1ed7b0085c30000f3, [2916]
2302
246 DATA ed73?085310085fde5dde5e5d5c5d9e5, [2729]
3516
247 DATA d5c5018e7fd9f508f508ed5fd61847ed, [1566]
3042
248 DATA 574fc5ed7be884ed56c9a7c0f3d9ed43, [2674]
2952
249 DATA f084d9fbc9fe02c0dd6603dd6e021a77, [1575]
1990
250 DATA c9131afeb038183a4785a72012219089, [2509]
2058
251 DATA 010000c5e536ed231acba7772318391b, [2988]
1766
252 DATA 1a1813fe02c0dd6603dd6e02e521eb84, [2062]
2050
253 DATA 341afeed28cbfec03065473a45854f87, [2847]
1756
254 DATA 87383079e6033c4f78fe760600281721, [2083]
1608
255 DATA 9089c5e5ebedb0eb36c32311c7877323, [2386]
2978
256 DATA 72d1cd9d87c1e1e75f23e757eb09eb72, [2614]
2608
257 DATA 2b73c9780600fe10280c328888cd8a89, [2244]
1362
258 DATA 180a0e0218e021f7843528f6131a4f87, [2415]
1584
259 DATA 300105e10303cd6f883eff324185c93a, [2632]
2258
260 DATA 458547e6033c4f2104857efe432842fe, [3098]
1976
261 DATA 522869fe4a2092780600873019 1a1332, [2212]
1504
262 DATA cd88cd8a89c3d5880103001899e11a77, [2479]
2452
```

CP/M Floppy Software Hardware Tips & Tricks Joyce

```
263 DATA 13231a7718c31a0ebdfee928054f21eb, [2354]
1728
264 DATA 843479e6600f0f0f0f4f21f88409eb18, [1886]
1278
265 DATA dc3a4685a7c25388237efe4120f71a13, [1979]
2102
266 DATA 320e89cd8a89cd138918bde113132a00, [2685]
1692
267 DATA 852b722b732200851b1b18b1237efe53, [1669]
1440
268 DATA 1a2842fecbca5188feed2821328589cd, [2544]
1822
269 DATA 8a893e02cd85894f3dca6e882a00855e, [3601]
2058
270 DATA 235623220085e1732372c3a188131a1b, [2504]
988
271 DATA fe452805fe4cc25188328989cd888921, [1882]
1260
272 DATA eb843418d7fecf2805feefc251883ecd, [2751]
2454
273 DATA 32c487cd9d873ec332c48718bfc93dc9, [2471]
2344
274 DATA ed002aee84e5f1c9,1032              [1357]
```

Programm BASPROG

```
100 '*****************                       [528]
110 '* MCode-TRACER *                         [1366]
120 '*****************                       [528]
130 '*   (c) 1985     *                      [439]
140 '* by Th. Fippl   *                      [570]
150 '*                *                      [1570]
160 '*  6336 Solms    *                      [593]
170 '*****************                       [528]
180 '                                        [117]
1000 CLEAR:DEFSTR a-k:DEFINT m-z             [2257]
1010 DEF FNp(m)=UNT(PEEK(m)+256*PEEK(m+1))   [1163]
1020 DEF FNh(m)=HEX$(FNp(m),4)               [1300]
1030 DEF FNb(m)=HEX$(PEEK(m),2)              [1434]
1040 DEF FNm(p,x)=UNT(p+x)AND &FFFF          [1343]
1050 cmd="DZSA":GOSUB 1810                   [939]
1060 PRINT#2,"> ";                           [328]
1070 LINE INPUT#2,i:i=UPPER$(i)              [1910]
1080 s=INSTR(1-4*(i=""),cmd,i)               [2169]
1090 ON s GOSUB 1120,1200,1340,1500          [1366]
1100 GOTO 1060                               [311]
1110 :                                       [174]
1120 p=0:INPUT#2,"Ausgabe per Drucker (j/n [2322]
)";i
1130 w=1-7*(LOWER$(i)="j")                   [585]
1140 GOSUB 1680                              [907]
1150 CLS#1:FOR n=1 TO 23                     [1194]
1160 GOSUB 1720:NEXT                         [1097]
1170 i=INKEY$:IF i=""GOTO 1170               [900]
1180 IF i=CHR$(13)GOTO 1150 ELSE RETURN      [2030]
1190 :                                       [174]
1200 INPUT#2,"CALL als Sprungbefehl   (j/n) [2629]
";i
1210 POKE &8546,-(LOWER$(i)<>"j")            [1182]
1220 INPUT#2,"Blockbef. im Stepmodus (j/n) [3413]
";i
1230 POKE &8547,-(LOWER$(i)<>"j")            [1677]
1240 GOSUB 1680:PRINT#1:w=1                  [1268]
1250 p=pc:GOSUB 1720                         [1892]
1260 pc=p:GOSUB 1910                         [1362]
1270 IF bef="???"THEN RETURN                 [1828]
1280 i=INKEY$:IF i=""GOTO 1280               [822]
1290 IF i<>CHR$(13)THEN RETURN               [2512]
1300 |TST,@pc,pc                             [979]
1310 IF PEEK(&8541)=&FF THEN PRINT#1         [1481]
1320 GOTO 1250                               [361]
1330 :                                       [174]
1340 CLS#5:PRINT#5,"Wert/Bank 0/Bank 3";     [3357]
1350 PRINT#5,"255     RAM      RAM           [1148]
1360 PRINT#5,"254     ROM      RAM           [2087]
1370 PRINT#5,"253     RAM      BASIC         [929]
1380 PRINT#5,"252     ROM      BASIC         [1105]
1390 PRINT#5,"0-251 RAM  Exp.ROM";           [1999]
1400 PRINT#5,"        [Discrom: 7]";         [2790]
1410 INPUT#2,"ROM Select Byte";st            [2250]
1420 IF st<0 OR st>255 GOTO 1410             [1687]
1430 POKE &8544,st:n=MAX(st,&FB)-&F8         [1874]
1440 IF st<&FC THEN LOCATE#5,1,7:PRINT#5,U [3123]
SING CHR$(24)+"###"+CHR$(24);st;
1450 PRINT CHR$(23)CHR$(1);                  [1919]
1460 FOR m=1 TO 15 STEP 2                    [1049]
1470 MOVE 152,n*16-m:DRAWR 143,0             [843]
1480 NEXT:|SET:GOTO 1910                     [1792]
1490 :                                       [174]
1500 INPUT#2,"Register     ";i               [2280]
1510 s=LEN(i)+(RIGHT$(i,1)="'")              [1465]
1520 IF s<1 OR s>2 THEN RETURN               [1192]
1530 n=s*8:PRINT#2,USING"##";n;              [1640]
1540 INPUT#2,"-Bit Wert";l                   [1136]
1550 IF l<-2^(n-1)OR l>2^n-1 GOTO 1530       [3169]
1560 a=LEFT$(UPPER$(i)+" ",s+1)              [2937]
1570 m=UNT(l):IF s=2 GOTO 1620               [2101]
1580 l=(INSTR("I R F A'F A C'B'E'D'L'H'C B [3771]
 E D H L ",a)-1)/2
1590 IF l<>INT(l)THEN RETURN                 [1722]
1600 POKE &84EA+l,PEEK(@m)                   [1297]
1610 GOSUB 1910:GOTO 1500                    [1543]
1620 l=(INSTR("IR AF'AF BC'DE'HL'BC DE HL  [3262]
IX IY SP ",a)-1)/3
1630 IF l<>INT(l)THEN RETURN                 [1722]
1640 p=&84EA+l*2:POKE p,PEEK(@m)             [2125]
1650 POKE p+1,PEEK(@m+1)                     [682]
1660 GOSUB 1910:GOTO 1500                    [1543]
1670 :                                       [174]
1680 INPUT#2,"Adresse";l                     [1342]
1690 IF l<&8000 OR l>65535 GOTO 1680         [2255]
1700 pc=UNT(l):RETURN                        [2628]
1710 :                                       [174]
1720 PRINT#w,HEX$(pc,4)"   ";                [1323]
1730 bef="":|DIS,@bef,pc                     [1592]
1740 FOR z=0 TO PEEK(&8545)AND 3             [1146]
1750 m=0:|PEK,@m,pc:pc=FNm(pc,1)             [2652]
1760 PRINT#w,HEX$(m,2)" ";:NEXT              [1010]
1770 s=INSTR(3,bef+" "," ")                  [2357]
1780 PRINT#w,TAB(20)LEFT$(bef,s-1)TAB(24)M [2273]
ID$(bef,s)
1790 RETURN                                  [555]
1800 :                                       [174]
1810 MODE 2:BORDER 2:|SET                    [1927]
1820 FOR n=1 TO 5                            [862]
1830 READ bez,pn,x1,x2,y1,y2                 [1255]
1840 GOSUB 2080:NEXT                         [1604]
1850 PRINT#3," Z - Single Step               [1522]
1860 PRINT#3," D - Disassemble               [1835]
1870 PRINT#3," S - ROM Select                [1599]
1880 PRINT#3," A - Alter Reg.                [1589]
1890 WINDOW#6,14,17,20,24:GOTO 1340          [2086]
1900 :                                       [174]
1910 LOCATE#4,1,1                            [364]
1920 PRINT#4,"PC "HEX$(pc,4)"  IX "FNh(&84 [2583]
FC);
1930 PRINT#4,"SP "FNh(&8500)"  IY "FNh(&84 [3485]
FE)
1940 PRINT#4,"HL "FNh(&84FA)"  HL'"FNh(&84 [2735]
F4);
1950 PRINT#4,"DE "FNh(&84F8)"  DE'"FNh(&84 [2262]
F2);
1960 PRINT#4,"BC "FNh(&84F6)"  BC'"FNh(&84 [1996]
F0)
1970 PRINT#4,"A "FNb(&84EF)"   I "FNb(&84EA [2417]
)"  ToS:";
1980 PRINT#4,"A'"FNb(&84ED)"   R "FNb(&84EB [3284]
)
1990 LOCATE#4,3,11                           [476]
2000 PRINT#4,"SZ.H.PNC                       [639]
2010 PRINT#4,"F "BIN$(PEEK(&84EE),8)         [1436]
2020 PRINT#4,"F'"BIN$(PEEK(&84EC),8);        [1792]
2030 st=FNp(&8500):CLS#6                     [1503]
2040 FOR n=8 TO 0 STEP-2                     [887]
2050 PRINT#6,FNh(st+n);                      [1466]
2060 NEXT:RETURN                             [940]
2070 :                                       [174]
2080 PAPER pn:PEN 1-pn                       [1241]
2090 LOCATE x1-1,y1-1                        [712]
2100 PRINT CHR$(150)CHR$(24)bez;CHR$(24)ST [4632]
RING$(x2-x1-LEN(bez)+1,CHR$(154))CHR$(156)
2110 FOR y=y1 TO y2                          [1286]
2120 LOCATE x1-1,y                           [636]
2130 PRINT CHR$(149)                         [1036]
2140 LOCATE x2+1,y                           [987]
2150 PRINT CHR$(149)                         [1036]
2160 NEXT                                    [350]
2170 LOCATE x1-1,y                           [636]
2180 PRINT CHR$(147)STRING$(x2-x1+1,CHR$(1 [2312]
54))CHR$(153);
2190 WINDOW x1,x2,y1,y2                      [1672]
2200 CLS:WINDOW SWAP n                       [493]
2210 RETURN                                  [555]
2220 :                                       [174]
2230 DATA OUTPUT,1,40,79,2,24                [986]
2240 DATA INPUT,0,2,37,2,9                   [1080]
2250 DATA Helpscreen,0,20,37,12,15           [1736]
2260 DATA Z80 Register,1,2,17,12,24          [1726]
2270 DATA ROM Select,0,20,37,18,24           [903]
```

Auf einen Blick, alles im Griff!

Die CP/M-Referenzkarten beinhalten die wichtigsten und am häufigsten benötigten Befehle und Funktionen von CP/M 2.2. Sie sind übersichtlich aufgebaut, in Gruppen unterteilt und ersparen somit das mühsame Blättern im Handbuch!
Bei der Konzeption dieses Sonderheftes haben wir streng darauf geachtet, daß die Referenzkarten herausgetrennt und separat aufbewahrt werden können.
Im einzelnen werden folgende Befehle und Funktionen übersichtlich dargestellt:

CTRL-Befehle
DDT-Monitorbefehle
Stat-Kommando
Editor-Befehle
Save-Befehle
Direkt-Konsol-Befehle
Farben und Modes

Hinweis:
Die Referenzkarten sind so angeordnet, daß sie problemlos, durch Öffnen der Heftklammern, zur Aufbewahrung entnommen werden können!
(SR)

```
-------------------------------------------------------------------------
>> CP/M 2.2  allgemeine Hinweise <<
---------------------------------
-------------------------------------------------------------------------
I  MODE - Einstellung am CPC :                                          I
I                                                                       I
I        CTRLd0   'entspricht MODE 0       (CTRLd = ^d 'entspricht      I
I        CTRLd1   'entspricht MODE 1        gleichzeitigem druecken     I
I        CTRLd2   'entspricht MODE 2        der CTRL u. d-Taste)        I
I-----------------------------------------------------------------------I
I  FARB - Einstellung am CPC :                                          I
I                                          __> xy Farbwahl (INK)        I
I  CTRL \@ xy  'entspricht PAPER          :__>                          I
I  CTRL \A xy  'entspricht PEN            :  > x = erste  Farbe         I
I  CTRL \] xy  'entspricht BORDER         :  > y = zweite Farbe         I
I                                                                       I
I  (INK)  Farbe                           (INK)  Farbe                  I
I  @= 0   'Schwarz                        n=14   'Pastellblau           I
I  a= 1   'Blau                           o=15   'Orange                I
I  b= 2   'Hellblau                       p=16   'Rosa                  I
I  c= 3   'Rot                            q=17   'Pastellmagenta        I
I  d= 4   'Magenta                        r=18   'Hellgruen             I
I  e= 5   'Hellviolett                    s=19   'Seegruen              I
I  f= 6   'Hellrot                        t=20   'helles Blaugruen      I
I  g= 7   'Purpur                         u=21   'Limonengruen          I
I  h= 8   'helles Magenta                 v=22   'Pastellgruen          I
I  i= 9   'Gruen                          w=23   'Pastell-blaugruen     I
I  j=10   'Blaugruen                      x=24   'Hellgelb              I
I  k=11   'Himmelbau                      y=25   'Pastellgelb           I
I  l=12   'Gelb                           z=26   'leuchtend Weiss       I
I  m=13   'Weiss                          :                             I
I  :_____________________________________ :                             I
I  : entspricht Farbnummer + 64dec        :__> Farbe entsprechend dem zugeord- I
I  :                                      :__> neten Buchstaben waehlen I
I  :                                                                    I
I  :__>ACHTUNG ! Erste und zweite Farbe gleich waehlen,kein blinken.    I
I                                                                       I
I-----------------------------------------------------------------------I
I               CTRL\@l    = Bildschirm blinkt zwischen der             I
I Beispiel :                 Farbe Schwarz(@=0) und Gelb(l=12)          I
I PEN,BORDER    CTRL\acz   = PEN blinkt zwischen der Farbe              I
I oder PAPER                 Rot(c=3) und leuchtend Weiss(z=26)         I
I blinkend      CTRL]ui    = BORDER blinkt zwischen der Farbe           I
I                            Limonengruen(u=21) u. Gruen(i=9)           I
I-----------------------------------------------------------------------I
I Beispiel :                                                            I
I              CTRL\@aaCTRL\axxCTRL]aaCTRLd1  - MODE 1                  I
I Farbwahl      :       :       :____________ BORDER Blau               I
I wie unter     :       :____________________ PEN    Gelb               I
I Basic         :____________________________ PAPER  Blau               I
I-----------------------------------------------------------------------I
I Beispiel:     SETUP  'alle Fragen mit Y beantworten ausser Frage 2    I
I Standard      Sign - on string :                                      I
I Einstellung   ^ww^\a@@^]wwCP/M2.2-Amstrad Consumer Electronics plc^J^M I
I bei CP/M      Is this correct (Y/N) n'ENTER                           I
I               Enter new sign-on string:_^\@@@^\aoo^]@@^d1'enter       I
-------------------------------------------------------------------------
```

```
>> CP/M 2-2 Allgemeine Hinweise <<
----------------------------------

-------------------------------------------------------------
I  DIREKT KONSOL-BEFEHLE >                                  I
I  ---------------------                                    I
I-----------------------------------------------------------I
I DIR          listet den gesamten Inhalt der Diskette      I
I DIR b:       listet alle Dateien von Laufwerk B.          I
I DIR *.BAS    listet alle Dateien mit BAS                  I
I DIR TEST.BAS listet nur diese Datei soweit vorhanden      I
I-----------------------------------------------------------I
I ERA          dient zum löschen von Dateien im Inhaltsverz. I
I ERA PIP.COM  löscht die Datei PIP.COM                     I
I ERA *.DAT    löscht alle DAT.Dateien                      I
I ERA *.*      löscht alle Dateien                          I
I-----------------------------------------------------------I
I REN          ermöglicht die Umbenennung von bestehenden Da- I
I              teien.                                       I
I REN versuch.bas=test.bas  (neuer Name=alter Name)         I
I-----------------------------------------------------------I
I TYPE         stellt spezifizierte Dateien auf dem Bildsdchirm I
I              dar.                                         I
I type test.bas Programm test.bas wird dargestellt          I
I              Es lassen sich nur Dateien mit dem Index BAS I
I              oder ASM auflisten.                          I
I-----------------------------------------------------------I
I                                                           I
I SAVE         ist nur für spezielle Anwendungen vorgesehen. I
I              .........................................    I
I                                                           I
I              Es erlaubt die Sicherung von  Dateien  ab der I
I              Speicherstelle 100hex.                       I
I              Die zu sichernden  Seiten  werden als Parameter I
I              angegeben,wobei eine Seite 256 Bytes entspricht.I
I              Benutzt wird  'Save'  nach Änderung eines  ASM- I
I              Programms etc. oder in Verbindung mit dem DDT- I
I              Programm.                                    I
I                                                           I
I SAVE n b:filename.typ         ( b:=Laufwerk B )           I
I                                                           I
I              n = Zahl von 1 .. 255 = Anzahl der zu sichernden I
I                                      Seiten.              I
I                                                           I
I              1 = 256 Bytes = 2 Records = 16 Reihen zu 16Bytes I
I                                                           I
-------------------------------------------------------------
```

```
>> CP/M     SAVE Seitenzahl Hilfstabelle <<
-------------------------------------------

-------------------------------------------------------------
I  n                              I  n                      I
I  1 = 0100 - 01FF =  2 Rec       I  16 = 1000 - 10FF =  4 KB I
I  2 = 0200 - 02FF =  4 Rec       I  32 = 2000 - 20FF =  8 KB I
I  3 = 0300 - 03FF =  6 Rec       I  48 = 3000 - 30FF = 12 KB I
I  4 = 0400 - 04FF =  8 Rec       I  64 = 4000 - 40FF = 16 KB I
I                      (1 KB)     I                         I
I  5 = 0500 - 05FF = 10 Rec       I  80 = 5000 - 50FF = 20 KB I
I  6 = 0600 - 06FF = 12 Rec       I  96 = 6000 - 60FF = 24 KB I
I  7 = 0700 - 07FF = 14 Rec       I 112 = 7000 - 70FF = 28 KB I
I  8 = 0800 - 08FF = 16 Rec       I 128 = 8000 - 80FF = 32 KB I
I                      (2 KB)     I                         I
I  9 = 0900 - 09FF = 18 Rec       I 144 = 9000 - 90FF = 36 KB I
I 10 = 0A00 - 0AFF = 20 Rec       I 160 = A000 - A0FF = 40 KB I
I 11 = 0B00 - 0BFF = 22 Rec       I 176 = B000 - B0FF = 44 KB I
I 12 = 0C00 - 0CFF = 24 Rec       I 192 = C000 - C0FF = 48 KB I
I                      (3 KB)     I                         I
I 13 = 0D00 - 0DFF = 26 Rec       I 208 = D000 - D0FF = 52 KB I
I 14 = 0E00 - 0EFF = 28 Rec       I 224 = E000 - E0FF = 56 KB I
I 15 = 0F00 - 0FFF = 30 Rec       I 240 = F000 - F0FF = 60 KB I
I 16 = 1000 - 10FF = 32 Rec       I                         I
I          I           (4 KB)     I      I                  I
I          I                          vierte hexziffer      I
I          I                       a) Je nach hex n ablesen I
I        dritte hexziffer                                   I
I     b) Je nach hex n ablesen     ( n=Seitenzahl )         I
I-----------------------------------------------------------I
I Beispiel: AFFF   a)= 160  b)= 15 n=a+b  n=175             I
I                                                           I
I           14FF   a)=  16 + b)=  4        n= 20            I
I                                                           I
I           42FF   a)=  64 + b)=  2        n= 66            I
I                                                           I
I           DAFF   a)= 208 + b)= 10        n=218            I
I                                                           I
I===========================================================I
I                                                           I
-------------------------------------------------------------
```

```
>> CP/M     SAVE Seitenzahl Hilfstabelle <<
-------------------------------------------

-------------------------------------------------------------
I  n                          I  n                          I
I  1 = 0100 - 01FF =  2 Rec   I 17 = 1100 - 11FF = 34 Rec   I
I  2 = 0200 - 02FF =  4 Rec   I 18 = 1200 - 12FF = 36 Rec   I
I  3 = 0300 - 03FF =  6 Rec   I 19 = 1300 - 13FF = 38 Rec   I
I  4 = 0400 - 04FF =  8 Rec   I 20 = 1400 - 14FF = 40 Rec   I
I                     (1 KB)  I                    (5 KB)   I
I  5 = 0500 - 05FF = 10 Rec   I 21 = 1500 - 15FF = 42 Rec   I
I  6 = 0600 - 06FF = 12 Rec   I 22 = 1600 - 16FF = 44 Rec   I
I  7 = 0700 - 07FF = 14 Rec   I 23 = 1700 - 17FF = 46 Rec   I
I  8 = 0800 - 08FF = 16 Rec   I 24 = 1800 - 18FF = 48 Rec   I
I                     (2 KB)  I                    (6 KB)   I
I  9 = 0900 - 09FF = 18 Rec   I 25 = 1900 - 19FF = 50 Rec   I
I 10 = 0A00 - 0AFF = 20 Rec   I 26 = 1A00 - 1AFF = 52 Rec   I
I 11 = 0B00 - 0BFF = 22 Rec   I 27 = 1B00 - 1BFF = 54 Rec   I
I 12 = 0C00 - 0CFF = 24 Rec   I 28 = 1C00 - 1CFF = 56 Rec   I
I                     (3 KB)  I                    (7 KB)   I
I 13 = 0D00 - 0DFF = 26 Rec   I 29 = 1D00 - 1DFF = 58 Rec   I
I 14 = 0E00 - 0EFF = 28 Rec   I 30 = 1E00 - 1EFF = 60 Rec   I
I 15 = 0F00 - 0FFF = 30 Rec   I 31 = 1F00 - 1FFF = 62 Rec   I
I 16 = 1000 - 10FF = 32 Rec   I 32 = 2000 - 20FF = 64 Rec   I
I                     (4 KB)  I                    (8 KB)   I
I-----------------------------I-----------------------------I
I 33 = 2100 - 21FF = 66 Rec   I 49 = 3100 - 31FF = 98 Rec   I
I 34 = 2200 - 22FF = 68 Rec   I 50 = 3200 - 32FF =100 Rec   I
I 35 = 2300 - 23FF = 70 Rec   I 51 = 3300 - 33FF =102 Rec   I
I 36 = 2400 - 24FF = 72 Rec   I 52 = 3400 - 34FF =104 Rec   I
I                     (9 KB)  I                   (13 KB)   I
I 37 = 2500 - 25FF = 74 Rec   I 53 = 3500 - 35FF =106 Rec   I
I 38 = 2600 - 26FF = 76 Rec   I 54 = 3600 - 36FF =108 Rec   I
I 39 = 2700 - 27FF = 78 Rec   I 55 = 3700 - 37FF =110 Rec   I
I 40 = 2800 - 28FF = 80 Rec   I 56 = 3800 - 38FF =112 Rec   I
I                    (10 KB)  I                   (14 KB)   I
I 41 = 2900 - 29FF = 82 Rec   I 57 = 3900 - 39FF =114 Rec   I
I 42 = 2A00 - 2AFF = 84 Rec   I 58 = 3A00 - 3AFF =116 Rec   I
I 43 = 2B00 - 2BFF = 86 Rec   I 59 = 3B00 - 3BFF =118 Rec   I
I 44 = 2C00 - 2CFF = 88 Rec   I 60 = 3C00 - 3CFF =120 Rec   I
I                    (11 KB)  I                   (15 KB)   I
I 45 = 2D00 - 2DFF = 90 Rec   I 61 = 3D00 - 3DFF =122 Rec   I
I 46 = 2E00 - 2EFF = 92 Rec   I 62 = 3E00 - 3EFF =124 Rec   I
I 47 = 2F00 - 2FFF = 94 Rec   I 63 = 3F00 - 3FFF =126 Rec   I
I 48 = 3000 - 30FF = 96 Rec   I 64 = 4000 - 40ff =128 Rec   I
I                    (12 KB)  I                   (16 KB)   I
I-----------------------------I-----------------------------I
I                                                           I
-------------------------------------------------------------
```

```
CONTROL = '^c' '^e' '^h' '^j' '^h' usw.
-------------

CTRL 'C'   Dieses führt einen Warmstart aus.Das heißt,es wer-
           den interne Informationen bis zu einem vordefinier-
           ten Status wieder  hergestellt,ohne  die  Programme
           im Kernspeicher zu zerstören.
           >>  oder  Diskette beim Auswechseln aus einem oder
           mehrereren Laufwerken neu eintragen.
           >>  oder laufendes,transistentes Programm unterbre-
           chen um zur Kommandoebene zurück zu kehren.
------------------------------------------------------------------
CTRL 'E'   Dient dazu ein  langes Kommando  auf der nächsten
           Zeile auf  dem Bildschirm fortzusetzen.Dies bewegt
           den Cursor zum Anfang der nächsten Zeile.
           Wenn weitergeschrieben wird und danch ENTER gegeben
           wird,so wird die gesamte Eingabe als Einheit inter-
           pretiert.
------------------------------------------------------------------
CTRL 'H'   Damit kann  man das  letzte Zeichen  auf dem Bild -
           schirm löschen.(Es wirkt ähnlich Delete)
------------------------------------------------------------------
CTRL 'J'   Es wird benutzt,um ein Linienfeld zu erreichen.Dies
           braucht man  z.B. um unter PIP ein Line-Feed zu be-
           wirken.
------------------------------------------------------------------
CTRL 'M'   Es bewirkt das gleiche wie ENTER
------------------------------------------------------------------
CTRL 'P'   Damit wird der Drucker EIN-/AUS geschaltet.
           Alles was auf dem Bildschirm sichtbar ist wird auch
           auf dem Drucker ausgegeben.
------------------------------------------------------------------
CTRL 'R'   Damit  kann  eine  Zeile nochmals  sichtbar gemacht
           werden ohne sie einzugeben.
------------------------------------------------------------------
CTRL 'S'   Damit kann eine Bildschirmanzeige unterbrochen wer-
           den.Mit  drücken irgendeiner  Taste läuft  die Pro-
           grammanzeige weiter.
------------------------------------------------------------------
CTRL 'U'   Dies bewirkt,daß z.B. beim vertippen Cursor zum An-
           fang geht und Zeile neu eingegeben werden muß.
------------------------------------------------------------------
CTRL 'X'   Dies ist ähnlich  'CTRL U' aber mit dem Unterschied
           daß der Cursor zum Zeilenanfang geht ohne die schon
           geschriebenen Buchstaben zu löschen.
------------------------------------------------------------------
USER       dient  zur Trennung mehrerer Benutzer an einem Com-
           puter.Es können  USER-Befehle  0-15 gewählt werden.
           z.B. A>USER 1
------------------------------------------------------------------
```

```
>> DDT - MONITOR <<
------------------------------------------------------------------
'D' < Startadresse >,< Endadresse >
    Speicherinhalt ansehen.
    Listet in Hex und ASCII-Format
------------------------------------------------------------------
'L' < Startadresse >,< Endadresse >
    listet Speicherbereich in mnemonischer Form
    (Disassemble)
------------------------------------------------------------------
'A' < Adresse >
    wandelt mnemonische Befehle in Hex-Code um
    (Assemble)
------------------------------------------------------------------
'S' < Adresse >
    listet den Inhelt einer Speicherstelle und ermöglicht
    deren Änderung
------------------------------------------------------------------
'G' < Startadresse >
    startet ein Programm ab einer bestimmten Startadresse
------------------------------------------------------------------
'T' < Anzahl >
    arbeitet eine Anzahl von Programmschritten ab und
    zeigt dabei den aktuellen Prozessorstand an.
------------------------------------------------------------------
'U' < Anzahl >
    arbeitet eine Anzahl von Programmschritten ab,ohne
    aktuellen Prozessorstand anzuzeigen.
------------------------------------------------------------------
'X'   listet alle Register und Flags zusammen mit Inhalt
'X' < Register >
    dient zum Ändern eines Registers
------------------------------------------------------------------
'I' < Dateiname >
    Daten für die nächste Leseoperation wird vorgemerkt
------------------------------------------------------------------
'R' < Startadresse >
    liest eine Datei ab der Startadresse in den Speicher
------------------------------------------------------------------
'F' < Start >,< Ende >,< Wert >
    füllt einen Speicherbereich an eine bestimmte Ziel -
    Zieladresse
------------------------------------------------------------------
'M' < Start >,< Ende >,< Ziel >
    kopiert einen Speicherbereich an eine bestimmte Ziel-
    adresse.
------------------------------------------------------------------
'H' < Wert 1>,< Wert 2 >
    addiert und subtrahiert zwei Werte
------------------------------------------------------------------
```

```
>> STAT - Kommando <<
------------------------------------------------------------------
A>STAT     Antwort eventuell  A:R/W,SPACE: 14k
           STAT ist ein Programm das statistische Daten }ber
           eine Diskette je nach Parameter liefert.
------------------------------------------------------------------
A>STAT b: denkbare Antwort 'Bytes Remaining on B:27k'
          damit kann also der freie Speicherplatz abgefragt
          werden.
------------------------------------------------------------------
A>STAT ED.COM        Recs Bytes  Ext  Acc
                      52    7k    1   R/W A:ED.COM
                     Bytes Remaining On A: 14k
                     A>
------------------------------------------------------------------
           1 KB=1024 Bytes
           Sind also 49k frei so bedeutet dies,da" 49*1024
           Bytes=50176 Bytes frei sind bzw. 50176 Zeichen ab-
           gespeichert werden k|nnen.Pro Blatt ca. 3600 Zei-
           chen also 14 Blatt entspricht.
------------------------------------------------------------------
A>stat a:D*.com              (alle Programme mit D am Anfang)
          Recs  Bytes  Ext Acc
           38     5k    1  R/W A:DDT.COM
           19     3k    1  R/W A:DISCCHK.COM
           21     3k    1  R/W A:DISCCOPY.COM
            4     1k    1  R/W A:DUMP.COM
          Bytes Remaining on A: 14k
------------------------------------------------------------------
          Rec = wieviel Sektoren auf der Diskette von einem
                File benutzt werden.
                Jeder SEKTOR kann 128 Bytes speichern.
          1k  = 1024 Bytes        Ext = Erweiterungen
          ACC = Angabe des Attributs,filename und Status
------------------------------------------------------------------
A>STAT *.* $r/o              (wildcards=planlose Kartei)
PIP.COM set to R/O
SUBMIT.COM set to R/O        alle Dateien werden auf
etc .....                    Read only R/O gesetzt
RELAKOL.COM set to R/O
------------------------------------------------------------------
A>stat termy.bas $r/w      'TERMY.BAS set to R/W'
A>stat termy.bas $sys      'TERMY.BAS set to SYS'
A>stat termy.bas $dir      'TERMY.BAS set to DIR'
   >>Achtung .SYS Datei ist auf der Diskette nicht lesbar --
     R}cksetzung mit DIR<<
------------------------------------------------------------------
In CP/M gibt es vier logische Einheiten:
1. CON: Funktion der Operatorkonsole         (Bildschirm)
2. RDR: Funktion des Lochstreifenlesers      (auch RS232)
3. PUN: Funktion des Lochstreifenstanzers    (auch RS232)
4. LST: Funktion des Druckers
------------------------------------------------------------------
```

```
>> STAT <<

Entsprechend gibt es 12 physische Einheiten:
1. TTY: Langsame Konsolenanzeige (Fernschreiber)
2. CRT: Schnelle Konsolenanzeige (Bildschirm)
3. BAT: Einheit f}r Stapelverarbeitung
4. UC1: Vom Benutzer definierte Konsole
5. PTR: Lochstreifenleser
6. PTP: Lochstreifenstanzer
7. UR1: Leser #1
8. UR2: Leser #2
9. UP1: Stanzer #1
10. UP2: Stanzer #2
11. LPT: Zeilendrucker
12. UL1: Listeinheit
--------------------------------------------------
A>stat dsk:            (CP/M Diskette)

      A: Drive Characteristics
   1368: 128 Byte Record Capacity
    171: Kilobyte Drive  Capacity
     64: 32  Byte Directory Entries
     64: Checked  Directory Entries
    128: Records/ Extent
      8: Records/ Block
     36: Sectors/ Track
      2: Reserved Tracks
--------------------------------------------------
A>stat val:

Temp R/O Disk: d:=R/O
Set Indicator: d:filename.typ $R/O $R/W $SYS $DIR
Disk Status  : DSK: d:DSK
User Status  : USR:
Iobyte assign:             (Ist eine kleine Helpfunktion)
CON: = TTY: CRT: BAT: UC1:
RDR: = TTY: PTR: UR1: UR2:
PUN: = TTY: PTP: UP1: UP2:
LST: = TTY: CRT: LPT: UL1:
--------------------------------------------------
Achtung ':'der Doppelpunkt ist Bestandteil des Namens einer
         einer logischen oder physischen Einheit.

A>stat lst:=tty:          (Dabei gibt STAT keine Meldung aus,
A>                        um }ber die Zuordnung zu informieren)
--------------------------------------------------
A>stat usr:
Active User:  0    Die Meldung zeigt,da^ z.Z. im Benutzerbe-
Actives Files:0    reich 0 gearbeitet wird u. 0 Dateien ge-
                   speichert wurden.
```

```
>> EDitor-Befehle <<

'-' < Zeilenzahl >        springt eine bestimmte  Anzahl von
                          rückwärts.
    < Zeilenzahl >        wie vor aber vorwärts
--------------------------------------------------
'a' < Zeilenzahl >        lädt die angegebene Zeilenzahl in den
                          Speicher u. können gelistet werden.
'#a'                      wie vor aber alle Zeilen des Files
--------------------------------------------------
'b'                       Sprung zur ersten Zeile
'-b'                      Sprung hinter letzte Zeile
--------------------------------------------------
< Zeichenzahl > 'c'       Springt eine bestimmte Anzahl von
                          Zeichen vor.
< -Zeichenzahl > 'c'      wie vor aber Rückwärts
--------------------------------------------------
                    'd'   löscht das erste Zeichen der aktuellen
                          Zeile.
< -Zeichenzahl > 'd'      löscht eine Anzahl von Zeichen
--------------------------------------------------
'e'                       der editierte File wird abgespeichert
                          und das Programm beendet.
--------------------------------------------------
'f' < Text >              sucht einen Text ab der aktuellen Zei-
                          le.Wird der Text gefunden so erfolgt
                          ein Sprung zum Anfang der Zeile.
--------------------------------------------------
'h'                       Schreibt einen Speicher und springt
                          zum Anfang des File
--------------------------------------------------
'i' < Text                ab der aktuellen Zeile wird der einge-
                          gebene Text eingefügt.
'i' < Text > CTRL Z       wie vor,der alte Text wird angefügt
--------------------------------------------------
'j' < Text1 > CTRLz < Text2 > CTRLz < Text3 > CTRLz fügt den
                          Text2 nach Text1 bis zum Text3 ein.
--------------------------------------------------
'k'                       aktuelle Zeile wird gelöscht.
- < Zeilenzahl > 'k'      angegenene Zeilenzahl wird gelöscht.
--------------------------------------------------
  < Zeilenzahl > 'l'      Springt angegebene Zeilenzahl vorwärts
                          bzw '-' rückwärts.
--------------------------------------------------
'm'                       ermöglicht das Einfügen von Worten etc
               z.B. #mfspringtCTRLZ-7diSpringtCTRLZ
               >> sucht das Wort 'springt' löscht 7 Buchstaben
               und ersetzt das Wort nun mit 'Springt'. <<
               f=find i=insert
--------------------------------------------------
'n' < Text >              sucht den Text und füllt automatisch
                          den Speicher
```

```
>> EDitor <<

'o'                 Holt die zuletzt gespeicherte Version des
                    editierten Files in den Speicher (aktuelles
                    Programm wird überschrieben.
--------------------------------------------------
'p'                 Es werden 23 Zeilen angezeigt.
'-p'                wie vor rückwärts
--------------------------------------------------
'q'                 Der Editor wird verlassen,ohne den editier-
                    ten File abzuspeichern.
--------------------------------------------------
'r'                 Lese transferiertes File bzw. .LIB
                    Achtung! Wird vor das vorhandene File gelegt
--------------------------------------------------
's'                 Damit kann ein Buchstabe ausgetauscht wer-
                    den.
--------------------------------------------------
't'                 Volle Zeile wird angezeigt.
--------------------------------------------------
'u'
--------------------------------------------------
'v'                 Zeilennummer wird gesetzt
'-v'                an Bildschirmanfang ohne Zeilennummer
--------------------------------------------------
'w'                 Schreibt Text in den Speicher
< Zeilenz.>'w'      Schreibt Anzahl von Zeilen in den Speicher
--------------------------------------------------
'x'                 Speichert in X$$$$$$$.LIB File
--------------------------------------------------
< Anzahl > 'z'      Verzögerung je nach Anzahl
--------------------------------------------------
>> Liste der File-Indexe <<

.DAT     Datenfile
.DOC     Dokumentationsdatei (Anleitung zu Programmen)
.ASM     Mnemonische Datei
.HEX     Binär-Datei
.BAK     Sicherungsfile (wird beimzweiten Speichern angelegt
.TXT     Textdatei
.MAC     Makroassembler Datei
.PRN     druckreife Version eines Textes/Programms (listen
         auf den Drucker durch den Befehl PIP LST:=filename
.LIB     Bibliotheksdatei
.BAS     Basicfile
.BIN     Binär-Datei  wie HEX
.$$$     Zwischendatei
.COM     CP/M-File
```

VDOS 2.0

Die Firma Vortex hat seit dem Erscheinen der Schneider Computer ständig neue Peripherie für diese Rechner entwickelt. Neben den bekannten 5 ¼"-Laufwerken, der Speichererweiterung und der Festplatte, ist nun ein neues Betriebssystem entwickelt worden - das VDOS 2.0.

Da aber Vortex-Laufwerke schon über ein Jahr mit dem alten VDOS 1.0 ausgeliefert werden, bleibt die Frage offen, wie eben diese Besitzer in den Genuß des erweiterten DOS kommen. Vortex bietet dazu einen Service: das DOS wird bei Einsendung des alten VDOS gegen die neue Version ausgetauscht.

Einbau ohne Probleme

Vor dem Einbau des neuen EPROMs muß das alte 44K CP/M wegen der, auf 128 erweiterten, DIR-Einträge unter Anleitung des mitgelieferten, umfangreichen Handbuches mit dem Dienstprogramm SYSGEN um 256 Byte verkleinert werden. Anschließend wird das alte VDOS-EPROM vorsichtig mit einem kleinen Schlitzschraubenzieher aus dem Kontroller-Sockel herausgehebelt und das neue VDOS eingesetzt. Trotz des relativ leichten Einbaus, ist unbedingt zu beachten, daß die Garantie des Rechners erlischt, sobald ein Fremdeingriff vorgenommen wurde. Nach dem Einschalten des Rechners ist das neue Betriebssystem an einer veränderten Meldung zu erkennen. Ohne jedes Problem werden die vorher beschriebenen Disketten unter Basic gelesen und je nach Bedarf auf bis zu 128 DIR-Einträge erweitert, was den Nutzungsgrad noch einmal erheblich verbessert.
In diesem Zusammenhang wurde auch der CAT-Befehl modifiziert, indem mehr Informationen über die Verteilung des Speicherplatzes auf der Diskette ausgegeben werden. Will man unter CP/M arbeiten, so muß man lediglich von der verkleinerten Systemdiskette booten.
Beim Arbeiten mit der Speichererweiterung kann auch sofort das 62K CP/M geladen werden, welches ebenfalls bis zu 128 DIR-Einträge zur Verfügung stellt. Ebenso die als Laufwerk C ansprechbare RAM-Disk.

Hello Bas

Eine tolle Idee ist das neue, sogenannte »Turn-Key-System«. Bei jedem Einschalten oder Rücksetzen des Computers wird erst einmal nachgeschaut, ob sich nicht ein File mit Namen "Hello.bas" oder "Hello.bin" auf der Diskette befindet. In diesem File kann z.B. ein Programm stehen, welches wiederum ein anderes Programm aufruft und startet, ohne daß man einen Finger krumm machen muß. Ohne weiteres läßt sich auf diese Art und Weise auch CP/M booten.

Relative Dateiverwaltung

Wurde schon mit dem Betriebssystem BOS, das für die Verwaltung der Speichererweiterung unter Basic zuständig ist, eine relative Dateiverwaltung auf einer Pseudofloppy realisiert (s.a. CPC Nr. 2/86), so kommen jetzt auch Schneider-Besitzer ohne Speichererweiterung in den Genuß des relativen Dateizugriffs. Unter BOS können die neuen VDOS 2.0-Befehle natürlich auch angesprochen werden. Um der speziellen Speicherstruktur bei relativen Dateien gerecht zu werden, sind eine ganze Reihe von zusätzlichen Befehlen vonnöten, die als RSX-Calls in das neue Betriebssystem eingebunden wurden. Mit dem Befehl FILES wird ein Datenpuffer für einen der 16 möglichen Kanäle einer Datei definiert. Dieser Kanal wird dann mit einem modifizierten OPEN-Befehl geöffnet, wobei auch die Angabe der aktuellen Länge eines Records erfolgt. Die tatsächliche Strukturierung des mit FILES geschaffenen Puffers, also eines Records, geschieht mit dem Befehl FIELD. GET liest anschließend einen beliebigen Record von der Diskette ein und übergibt ihn entsprechend der FILES-Strukturierung an Stringvariablen. Beim Schreiben vollzieht sich die gleiche Prozedur, wobei an Stelle von GET der Befehl PUT gewählt werden muß. Die Befehle erinnern stark an das MBASIC von MICROSOFT (s.a. CPC Nr. 3/86, CP/M und seine Möglichkeiten), das ja gewissermaßen einen Standard für Basic darstellt. MBASIC faßt noch die Befehle FILES und FIELD zusammen und ermöglicht durch zusätzliche Befehle das Schreiben der Records als platzsparenden Binärcode, was an das FORMAT-freie Schreiben unter FORTRAN erinnert.
Der Vorteil von relativen Dateien gegenüber den bekannten sequentiellen, liegt in der Tatsache, daß nicht die gesamte Datei der Reihe nach gelesen werden muß, sondern ein beliebiger Datensatz über eine Record-Nummer, die mit dem Befehl GET bzw. PUT übergeben wird, bearbeitet werden kann. Das bringt natürlich erhebliche Zeitvorteile mit sich und ermöglicht damit unter Basic eine absolut professionelle Programmierumgebung.

Verbessertes ERA und REN

Muß an einen RSX-Befehl eine Variable vom Typ String übergeben werden, kann dies unter VDOS 2.0 auch direkt geschehen. Zum Löschen eines Files gibt es jetzt zwei Möglichkeiten:

1. **a$="name":|ERA,@a$**
2. **|ERA,"name"**

Dies gilt für alle RSX-Calls des Floppy-Controllers und der Speichererweiterung.

Abfangen von Diskettenfehlern

Nach Eingabe des neuen RSX-Befehls DERROR können auch Diskettenfehler vom Programm direkt abgefangen werden, ohne daß es zu einem Programmabbruch kommt. Abhängig von der Art und Weise des eingetretenen Fehlers, kann das Programm eine entsprechende Verzweigung vornehmen.
Im Zuge der relativen Dateiverwaltung sind außerdem eine ganze Reihe von neuen Fehlermeldungen hinzugekommen.

Die verschnellerte Bildschirmausgabe

Durch eine Reihe weiterer RSX-Calls ist es unter anderem möglich, die Bildschirmausgabe zu verschnellern. Der einzige Nachteil, der dabei in Kauf genommen werden muß, liegt in der Tatsache, daß dabei die Window-Technik nicht mehr unterstützt werden kann. Dieser FAST-Befehl ist übrigens auch unter CP/M ablauffähig und kann zum Beispiel dazu verwendet werden, die Geschwindigkeit der Bildschirmausgabe unter WORDSTAR (s.a.CPC Sonderheft Nr. 2) zu verbessern. Sechs weitere Befehle unterstützen die Erzeugung von Grafik. Unter CP/M wird der Laufwerksmotor jetzt auch nach der »Schreibschutz«-Meldung ausgeschaltet, so daß die Diskette entfernt und/oder ausgewechselt werden kann.

Der ROM-residente Monitor

Der eingebaute Monitor, mit dem Maschinenspracheprogramme geladen, getestet und wieder abgespeichert werden können, ist ein leistungsfähiger Tool, der sofort und ohne wesentlichen Speicherplatz zur Verfügung steht. Er beinhaltet viele Funktionen, wie z.B. Disassemblieren, Anzeigen und Ändern von Speicher- und Registerinhalten, Setzen von Breakpoints, Einzelschrittausführung etc. Der Monitor kann sowohl unter Basic als auch unter CP/M aufgerufen werden. Als Nachteil ist zu beachten, daß keine Labels verarbeitet werden können. Die Programme PARA und GRAFIKMASTER sind erst nach dem Patchen lauffähig. Dazu muß mit DDT ein kleines Programm eingegeben werden. Zusätzlich erhält man dann auch noch eine kopierfähige Version von PARA.

Dipl.-Ing. H. Scheruhn

CP/M · Floppy · Software · Hardware · Tips & Tricks · Joyce

Mehr als nur von A – Z Disketten-Dateinamen

Das Problem:

Angenommen, ich finde in »Schneider CPC International« eine pfiffige kleine Routine, wie z.B. den deutschen Zeichensatz, eine Funktionstasten-Belegung oder einen selbstdefinierten Zeichensatz für Sonderanwendungen.

Meist gibt der Erfinder seinem Hilfsprogramm einen passenden Namen wie "DEUTZEI", "FUNKTAST" oder "SDZ". Zwar kann ich dieses Programm dann als erstes auf die neue Diskette speichern, um es immer verfügbar zu haben; aber wenn ich erst einmal 20 andere Programme oder Dateien dazu gespeichert habe, kann ich nur noch mit Mühe feststellen, welche Hilfsprogramme auf der Diskette vorhanden sind, weil etwa "SDZ" sich in der CAT-Liste irgendwo zwischen "RUNDLAUF" und "SUCHPIEK" unauffällig versteckt.

Hin und wieder gibt jemand die Scheinlösung aus: »Speichern Sie das Hilfsprogramm unter dem Namen "A", dann erscheint es als erstes in der CAT-Liste.«

Schön und gut – aber wenn ich je zwei oder drei solcher Hilfsprogramme auf verschiedenen Disketten in verschiedenen Kombinationen brauche?

Die Lösung:

Das Ei des Kolumbus liegt darin, daß AMSDOS (also das Disketten-Betriebssystem der Schneider CPC-Typen) für Dateinamen über die Buchstaben A – Z hinaus auch eine gar nicht so kleine Anzahl von anderen Zeichen akzeptiert, nämlich:

# (ASCII 35)	0 (ASCII 48)	↑ (ASCII 94)
$ (ASCII 36)	1 (ASCII 49)	` (ASCII 96)
& (ASCII 38)	2 (ASCII 50)	
' (ASCII 39)	3 (ASCII 51)	
+ (ASCII 43)	4 (ASCII 52)	
– (ASCII 45)	5 (ASCII 53)	
	6 (ASCII 54)	
	7 (ASCII 55)	
	8 (ASCII 56)	
	9 (ASCII 57)	

Was bedeutet das praktisch?
Mein selbstgestrickter deutscher Zeichensatz auf deutscher Tastatur heißt jetzt einfach "#". Und der CAT-Befehl bringt ihn jetzt immer an der ersten Stelle der Liste, weil das Gatter "#" den niedrigsten ASCII-Wert überhaupt hat, den AMSDOS akzeptiert!

Andere Hilfsprogramme heißen jetzt "+TAST" (für bequemes Programmieren in BASIC) und "+TEXT" (hausgemachtes Textverarbeitungsprogramm – primitiv, aber im praktischen Gebrauch gut bewährt). Diese erscheinen nun auf der CAT-Liste gleich hinter "#", noch vor den Programmen "AA" und "ABC".

Wer es etwas vornehmer haben will, könnte ebensogut den deutschen Zeichensatz als "0" bezeichnen, die Funktionstasten mit "1" und das Textprogramm mit "2". Der Effekt wäre im Prinzip derselbe (streng genommen kann man einen vergleichbaren Effekt über die USER-Funktion erreichen – aber eben viel umständlicher!).

Weitere Möglichkeiten:

Ein anderes Problemchen tritt auf, wenn ich beim Programmieren über eine kleine Teilroutine stolpere, die nicht so läuft, wie ich mir das vorgestellt habe, und die ich, isoliert vom Rest, durchprobieren muß.

Was tun? Ich speichere das unfertige Hauptprogramm unter "↑" ab (ASCII 94), so daß es auf der CAT-Liste hinter "ZYX" erscheint, und arbeite an der Teilroutine weiter. Muß ich auch diese zwischendurch speichern, heißt sie für mich vorübergehend "`". Das erleichtert später, wenn man es geschafft hat, das Löschen. (Der sonst so nett vorgeschlagene Name "TEST" verkriecht sich ebenfalls gern in der Liste.)

Affenstall läßt grüßen!

Im Prinzip läßt sich AMSDOS auch noch auf einige andere Zeichen ein:

@ (ASCII 64)
{ (ASCII 123)
} (ASCII 125)
~ (ASCII 126)

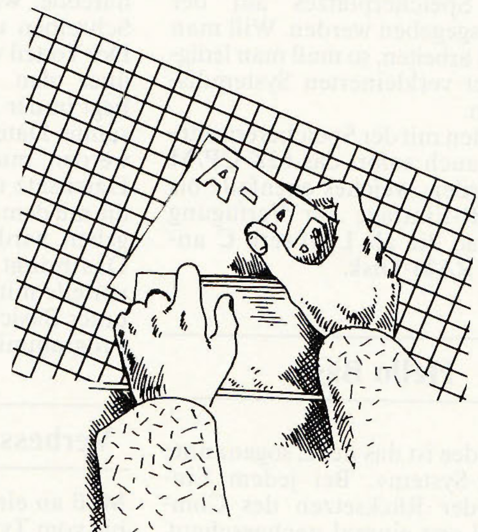
Wer grundsätzlich mit der QWERTY-Tastatur arbeitet, kann diese genauso benutzen, wie die oben aufgeführten anderen Zeichen.

Wer aber vorwiegend mit dem deutschen Zeichensatz arbeitet, sollte von diesen letzten vier Zeichen die Finger lassen. Denn weil der Drucker normalerweise erwartet, daß die deutschen Sonderzeichen mit ganz bestimmten ASCII-Zeichen getauscht werden (auf den ASCII-Werten 64, 123, 125 und 126 liegen dann zum Beispiel §, ä, ü, und ß), kann man die verdrängten ASCII-Zeichen nicht mehr erreichen. Das heißt, man kann schon, nur bringt die Bildschirmausgabe einen dann doch durcheinander.

Dies würde die gerade erst so schön übersichtlich gewordene CAT-Liste wieder unnötig verunstalten.

Ansonsten wäre natürlich ein Dateiname wie "@STALL" (sprich: Klammeraffenstall) schon für sich ein Spaß.

Martin Lipka
unter Mitarbeit von Thomas und Christoph Lipka



Data Media MAILORDER

FORTH

Abersoft (Nur Keyboardgebrauch)

Das Programm ermöglicht, daß das in der benutzten Sprache laufende Programm in der Geschwindigkeit eines Maschinen-Code-Programms läuft. Das Programm enthält eine Vielzahl möglicher Hardwarekonfigurationen bezüglich des CPC 464 und ist leicht zu benutzen.

Cassette Best.-Nr. 126 **89,– DM**

DEVPAC – ASSEMBLER/DISASSEMBLER

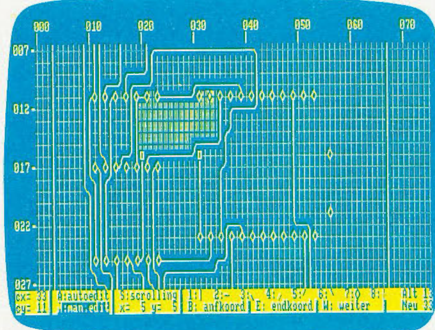
Hisoft (Nur Keyboardgebrauch)

Ein brauchbares Programm für Z-80 Assembler/Disassembler, ein Editor- und Monitor-„DEVPAC". DEVPAC erlaubt Ihnen, ein Maschinen-Code-Programm zu entwickeln, mit sämtlichen Anweisungen und Bedingungssätzen im 40 oder 80 Zeichen-Modus. Es ermöglicht Assembler ohne Freiräume, Listenausdrucke etc. Der Disassembler/Monitor erlaubt die Besichtigung des Speicherinhalts im 80 Zeichen-Modus.

Cassette Best.-Nr. 128 **99,– DM** Diskette 3" Best-Nr. 228 **129,– DM**

PLATINENKIT (Data Media)

Erstellt Layout von elektrischen Schaltungen. Einfachste Handhabung. Eingabe der Bauteile und Verbindungen vom User. Der Computer berechnet und zeichnet die korrekten Leiterbahnführungen. Ausgabe des Platinenplans sowie der Lötstopmaske auf Drucker.



Cassette Best.-Nr. 140
189,– DM

Diskette 3" Best.-Nr. 240
199,– DM

Diskette 5¼" Best.-Nr. 340
199,– DM

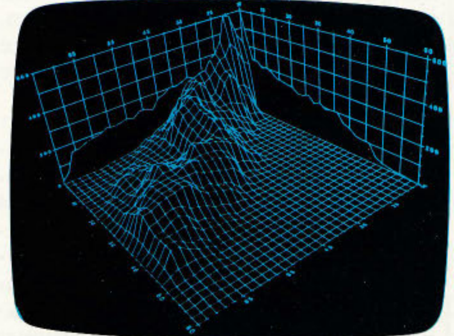
SYNTHESIZER (Data Media)

Neuartige Software! Ihr Schneider-Computer wird zu einem modernen Synthesizer. Die Klangfülle und Harmonie ist unübertroffen. Einen Sonderprospekt können Sie mit Einsendung eines frankierten Rückumschlages anfordern.

Cass. Best.-Nr. 154 **89,– DM** Disk. 3" Best.-Nr. 254 **99,– DM** Disk. 5¼" Best.-Nr. 354 **99,– DM**

FOCUS (Data Media)

Programm zur perspektivischen Darstellung von 3D-Grafiken. Stellen Sie z.B. ein Haus dar und verändern Sie durch Eingabe der Winkel die Perspektive!



Cassette Best.-Nr. 138
79,– DM

Diskette 3" Best.-Nr. 238
89,– DM

Diskette 5¼" Best.-Nr. 338
89,– DM

MULTIDATEI (Data Media) Ein universelles Dateiverwaltungsprogramm.

Cass. Best.-Nr. 141 **89,– DM** Disk. 3" Best.-Nr. 241 **99,– DM** Disk. 5¼" Best.-Nr. 341 **99,– DM**

ROUTINEN, TIPS UND TRICKS (Data Media)

Ca. 50 wichtige Programmroutinen für jeden Programmierer, wie z.B. Sortierroutine, Inkeyroutine, Diskettenroutine usw.

Cass. Best.-Nr. 151 **79,– DM** Disk. 3" Best.-Nr. 251 **89,– DM** Disk. 5¼" Best.-Nr. 351 **89,– DM**

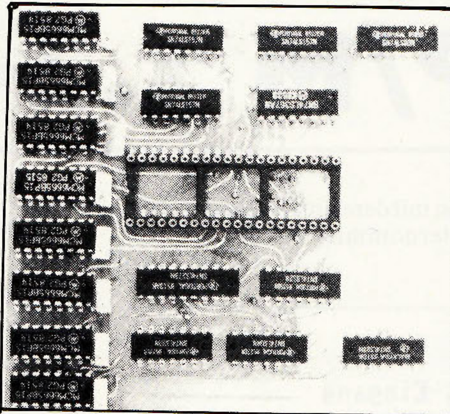
GREDI (Data Media)

Hilfsprogramm zur Herstellung von Hires-Grafiken.

Cass. Best.-Nr. 142 **49,– DM** Disk. 3" Best.-Nr. 242 **59,– DM** Disk. 5¼" Best.-Nr. 342 **59,– DM**

Das bringt Ihren Schneider CPC 464/664 auf Trab!

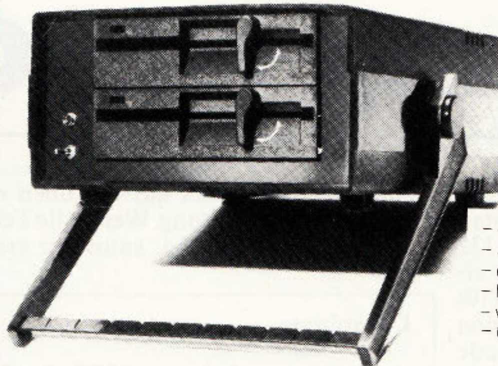
Speichererweiterung



- kompatibel zu CP/M 2.2, Schneider Basic
- voll kompatibel zu Laufwerken FDD und FDI
- eigenständiges Bank-Select
- erschließt die Welt der CP/M-Software
- eingebauter Bank-Manager
- einfache Montage

Preise: Data Media Speichererweiterung:

Best.-Nr. 6174	64KB Erweiterungsplatine, nicht aufrüstbar	**128,– DM**
Best.-Nr. 6170	64KB Erweiterung aufrüstbar	**198,– DM**
Best.-Nr. 6171	128KB Erweiterung aufrüstbar	**298,– DM**
Best.-Nr. 6172	256KB Erweiterung aufrüstbar	**398,– DM**
Best.-Nr. 6173	512KB Erweiterung komplett	**498,– DM**

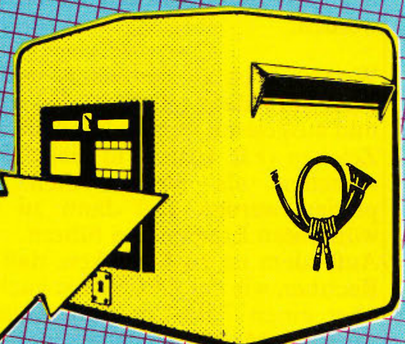


- professionelles Speichermedium im gängigen 5 1/4"-Format
- optional mit 2, 3 oder 4 Laufwerken zu je 780KB pro Laufwerk
- eigenständiges FDOS-Betriebssystem mit zusätzlichen Befehlen
- wahlweise 40 oder 80 Tracks
- CP/M 2.2 kompatibel
- eingebauter Maschinensprache-Monitor auf Eprom
- 72 Directory-Einträge à 20 Zeichen Länge
- eingebaute RS-232-Schnittstelle

5 1/4"-Laufwerk FDD

- eigener Controller
- Slimline Laufwerke (Teac 2x80 Tracks)
- eingebautes Netzteil
- hohe Laufruhe
- veränderbare Disc-Formate zum Lesen von Fremddisketten
- umfangreiches deutsches Handbuch

Anz. der Laufw.	Speicherkapaz.	Preis
2	1.6 MB	**1598,– DM**
3	2.4 MB	**2298,– DM**
4	3.2 MB	**2998,– DM**



Data Media MAILORDER

Bestellen Sie noch heute, Lieferung erfolgt umgehend! Zahlung per Vorkasse oder Nachnahme zuzügl. Porto- bzw. Nachnahmegebühr (Nachnahme ins Ausland ist nicht möglich).

Weitere Artikel in unserem Gesamtkatalog. Bitte anfordern (2,– DM für Rückporto beilegen).
Data Media GmbH -Mailorder- Ruhrallee 55, 4600 Dortmund, Tel.: (02 31) 12 50 71-3

Dokumentation

Viele Anwender haben sich ein CPC-System zugelegt, um für Zuhause ein preiswertes CP/M 2.2-System zu haben, auf dem sie bereits vorhandene Software nutzen können. Genau dies soll diese billige Softwarelösung ermöglichen.

Dieses Programmpaket wurde auf einem CPC 664 und einem APPLE II entwickelt. Es ist auch auf dem CPC 464 lauffähig. Eine Kopplung kann mit diesen Programmen an jeden CP/M 2.2-Rechner erfolgen, der über eine Centronics-Schnittstelle und einen funktionierenden CP/M 2.2-Basic-Interpreter (hier M-Basic) verfügt, der eine relative Dateiverwaltung erlaubt.

Die Idee

Der CPC simuliert für den Host-Rechner einen Drucker, das heißt, er übernimmt die Daten, die der andere Rechner zum Drucker schicken will.

Zu überwindende Schwierigkeiten:

Solange nur ASCII-Zeichen im Bereich zwischen 32 und 127 übermittelt werden, bestehen fast keine Schwierigkeiten. Wenn der Host-Rechner



Kopplung von CP/M 2.2-

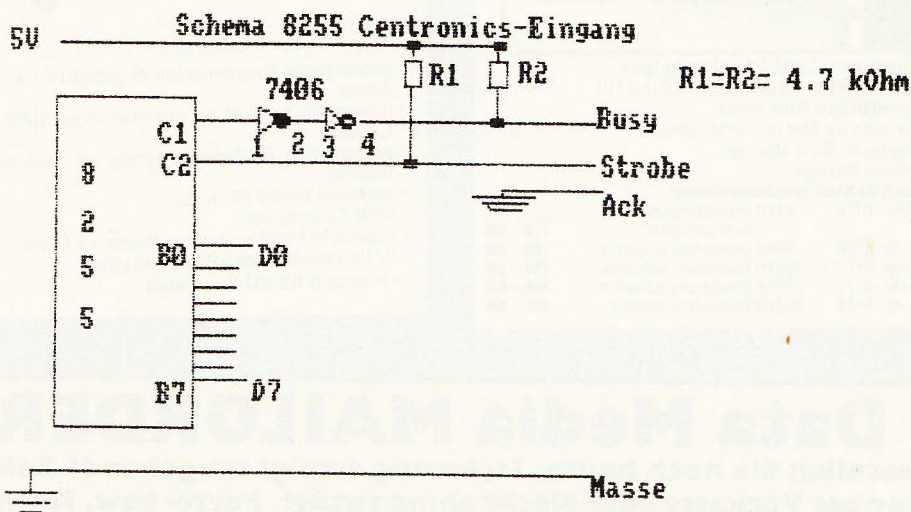
über eine Hochsprache betrieben wird, muß man bedenken, daß die meisten Hochsprachen nach spätestens 132 Zeichen einen Wagenrücklauf und einen Zeilenvorschub senden (chr$(10) und chr$(13)). Dies ist bei Textdateien höchstens störend, bei Maschinencode führt es zu schwerwiegenden Fehlern.

Weiterhin ist zu bedenken, daß auf vielen Rechnern die ASCII-Zeichen zwischen 0 und 31 nicht richtig an den Drucker übermittelt, sondern im Rechner als Steuerzeichen interpretiert werden.

Weiterhin ist zu bedenken, daß eine Maschinendatei nur relational eröffnet und ausgelesen werden kann, da einige Zeichen (z.B. Crtl-Z) in vielen Hochsprachen als Steuerzeichen interpretiert werden und dann zu unerwünschten Ergebnissen führen.
Außerdem ist zu bedenken, daß viele Rechner, wie der CPC selbst auch, nur über einen 7-Bit-Centronics-Ausgang verfügen, Maschinencode aber die vollen 8 Bit eines Bytes nutzt.

Alle Teile existieren aus der oben erwähnten Bauanleitung. Wenn alle Teile zusammengefügt sind, kann der erste Test mit dem untenstehenden Programm unternommen werden.

1. Hardware



Die Anschlüsse sind nach der obenstehenden Skizze herzustellen.



Rechnern

Hierfür wird das Druckerkabel des CPC mit dem neugebastelten Centronics-Eingang verbunden.

Programm: SELBSTEST

```
10 REM Programm Selbstest
20 i=-1:WIDTH 255
30 EVERY 10 GOSUB 130
40 MODE 2
60 b=&F8FD
70 c=&F8FE
80 steuer=&F8FF
90 OUT steuer,&X10000110
100 IF INP(c) AND 2 THEN 110
    ELSE 100
110 a=INP(b)
120 PRINT TAB(0);BIN$(a,8);" ";
    a;:GOTO 100
130 IF INP(c) AND 2 THEN RETURN
    ELSE i=i+1:IF i>128 THEN i=0
140 PRINT#8,CHR$(i);:RETURN
```

Dieses Programm nutzt die Interruptfähigkeit des Schneider CPC. Alle zehn Takte gibt der Rechner, wenn möglich, ein Zeichen auf den Druckerausgang, die über den Port im Hauptprogramm wieder eingelesen werden. Da der CPC nur über einen 7-Bit-Druckerausgang verfügt, können nur Zahlen zwischen 0 und 127 übernommen werden.

Der Portbaustein 8255 wird für dieses Programm im Mode 1 betrieben. Dies bedeutet, daß der Port A und der Port B als Eingangsports fungieren, der Port C für beide Steuerleitungen zur Verfügung stellt. Diese Steuerleitungen werden von dem 8255 selbst verwaltet. Für den Port B sind dies die Leitungen C1 und C2, wobei der Port C von C0 bis C7 reicht. Mit einem Signal auf C2 zeigt ein externes Gerät an, daß ein gültiges Datenbyte an Port B anliegt, dies entspricht dem STROBE-Signal des Centronics-Druckerausgangs. An C1 liegt solange ein High-Signal an, wie der Port B nicht in der Lage ist, ein Datenbyte anzunehmen. Erst wenn dieses Signal auf Low liegt, kann ein neues Datenbyte auf den Port B gelegt werden. Dies entspricht dem BUSY-Signal der Druckerschnittstelle. So stellt der 8255 intern sehr komfortabel eine Centronics-Eingangsschnittstelle zur Verfügung.
Wenn alles funktioniert, sollte auf dem Rechner eine aufsteigende Reihe der Bitmuster zwischen 1 und 127 erscheinen.
Wenn bis hier alles funktioniert, so kann man an die Rechnerkopplung gehen.

Software zur Rechnerkopplung

a) Übertragung von ASCII-Dateien

Aufbauend auf dem Programm Selbsttest kann man sich sehr schnell ein Programm zur Übermittlung von ASCII-Files schreiben. Dies könnte auf der Seite des CPC etwa so aussehen:

```
10 REM Programm ASCII.EMP
20 MODE 2
30 b=&F8FD
40 c=&F8FE
50 steuer=&F8FF
60 OUT steuer,&X10000110
70 IF INP(c) AND 2 THEN 80 ELSE
   70
80 a=INP(b)
90 IF a>31 AND a<127 THEN PRINT
   CHR$(a);
100 GOTO 70
```

Der Einfachheit halber gehe ich hier davon aus, daß auf dem Host-Rechner ein BASIC-Dialekt unter CP/M 2.2 zur Verfügung steht, in diesem Fall MBASIC (BASIC80).
Zum ersten Test reicht folgendes Programm:

```
10 REM Programm ASCII.SEN
20 a$=INKEY$ : IF a$="" THEN 20
30 LPRINT a$;
40 GOTO 20
```

Bei der Kopplung der Rechner sollten Sie daran denken, daß das Druckerkabel nicht länger als 2 m sein sollte, da man sonst mit einer steigenden Fehlerrate in der Übermittlung rechnen muß. Wenn die Verbindung hergestellt ist und beide Programme gestartet wurden, muß jedes Zeichen, das man auf dem Host-Rechner schreibt, auf dem Bildschirm des CPC erscheinen.

Übertragung von Maschinencode-Dateien

b1) Theorie

Jetzt soll ein Programmpaar vorgestellt werden, das zur Übertragung von Maschinencodedateien geeignet ist. Es ist vollständig auf Sicherheit ausgelegt und deshalb nicht sehr schnell.
Folgende Probleme und ihre Lösung sollen besprochen werden:

1. Die meisten Rechner besitzen nur eine 7-Bit Druckerschnittstelle. Maschinencode benutzt alle Bytewerte zwischen 0 und 255. Deshalb wird in diesem Programmpaket jedes Datenbyte in zwei Nibbles (4-Bit-Gruppen) übertragen, Bit 8 wird nicht genutzt, so daß sowohl 8-Bit-Schnittstellen oder 7-Bit-Schnittstellen benutzt werden dürfen.

2. Die Steuerzeichen mit den ASCII-Werten 0 bis 32 werden nicht immer richtig an den Drucker übergeben, der ASCII-Wert 27 dient auf den meisten Systemen z.B. zum Einleiten einer Steuersequenz für den Drucker. Das Programm umgeht diese Schwierigkeit, indem das Bit 8 permanent auf HIGH gesetzt wird. Dadurch können keine Steuerzeichenwerte im Übermittlungsprotokoll vorkommen.

3. Die Datenübertragungssicherheit ist bei Parallelschnittstellen relativ schlecht. Zum einen ist der Spannungsunterschied zwischen logisch 0 und logisch 1 viel geringer als bei V24-Schnittstellen, zum anderen ist die Abschirmung von zehn Leitungen gegen Fremdspannungen schwerer zu bewerkstelligen, als bei zwei Leitungen wie bei seriellen Schnittstellen.

b2) Das Übertragungsprotokoll

Übertragen wird die Datei in Blöcken von je 128 Bytes. Da auf Grund der Diskettenverwaltung von CP/M jede Datei aus einer ganzen Zahl von Gruppen von 256 Bytes besteht, ist im voraus die Anzahl von Blöcken zu ermitteln.

Der Block wird in vier Unterblöcke eingeteilt, das entspricht also Gruppen von 32 Informationsbytes.

Jedes dieser 32 Informationsbytes wird in zwei Nibble (4-Bit-Gruppen) zerlegt und dann gesendet. Zu diesen zwei

Bytes kommt noch die Adresse im Unterblock, die führend in einem Byte übermittelt wird.
Am Anfang und am Ende eines Unterblocks wird ein Anfangs- bzw. ein Endebyte als Kennzeichen gesendet. Hierbei dient das Anfangsbyte gleichzeitig als Synchronisationsbyte. Das gleiche gilt für den Anfang und das Ende eines Blocks. Insgesamt werden 32 Blöcke als eine Gruppe übermittelt, auch dafür gibt es ein Eingangs- und ein Ausgangskennzeichen. Schließlich gibt es noch ein »Ende der Datei«-Kennzeichen.

Das Sendeprogramm:

In den Data-Zeilen werden die genauen Namen der zu übertragenden Dateien abgelegt, danach die Gesamtanzahl der 128 Byte-Blöcke (mit dem STAT-Kommando zu erfahren) und der Startpunkt. Es ist damit auch möglich, eine sehr lange Datei in mehreren Teilstücken zu überspielen. Diese Datei wird als relative Datei eröffnet. Die Variable record$ beinhaltet die 128 Informationsbytes. Je 32 128 Byte-Blöcke werden zusammen übertragen, dann macht das Sendeprogramm eine Pause, um dem CPC das Ausscheiden der redundanten Bytes und das Auslagern der Informationen auf Diskette zu ermöglichen.

Das eigentliche Senden, d.h. "Drucken", der Zeichen verläuft in mehreren geschachtelten Schleifen. Die äußerste Schleife (Variable Q) bestimmt die zu sendenden 32 Blöcke, die Variable K zeigt an, welcher Unterblock bearbeitet wird, die Variable J beinhaltet die Adresse des gerade aktuellen Informationsbytes in diesem Unterblock.

Mit der Variablen R im Druckunterprogramm in Zeile 420 wird festgelegt, wie oft jedes Byte auf dem Drucker ausgegeben wird. Normalerweise reicht der Wert 2 vollkommen aus, bei mir lief das Programm auch ohne Wiederholung fehlerfrei. Nach dem Abschluß der Übertragung wird das Ende-Zeichen gesendet, dann geht das Programm in eine Warteschleife, bis der CPC die Datei ausgewertet hat (abhängig von der Dateilänge). Nun kann die nächste Datei gesendet werden (in den DATA-Zeilen festgelegt). Ist hierbei die Länge der zu sendenden Datei (Variable Zahl1) =0, dann bricht das Programm ab.

Die Programme sind ganz auf Übertragungssicherheit ausgelegt. Es ist möglich, mehrere Programme hintereinander zu übertragen, ohne daß dafür weitere Anweisungen durch den Bediener erforderlich sind. Durch die Wahl geeigneter Verzögerungsschleifen ist es theoretisch sogar möglich, ohne das Busy-Signal zu vollständigen Übertragungen zu kommen. Diese Verzögerungsschleifen müssen an den betreffenden Rechner angepaßt werden.

So sieht das Protokoll für einen Block aus:
Bit 8 ist immer high (=1):

Byte (Dual)	HEX	Bedeutung
10110100	B4	Anfangskennzeichen und Synchronisation
1011....	A.	Anfang Unterblock 1-4
1100....	C.	Adresse des Informationsbytes im Unterblock 1-32
1000....	8.	Low Nibble Informationsbyte
1001....	9.	High Nibble Informationsbyte
1100....	C.	Adresse des Informationsbytes im Unterblock 1-32
1000....	8.	Low Nibble Informationsbyte
usw.		
10111111	BF	Ende der Datei Kennzeichen

Punkte kennzeichnen variable Werte, die durch die Adresse oder das Informationsbyte geliefert werden. Diese Werte werden ausschließlich im Low Nibble übergeben.

Das Sendeprogramm für den Host - Rechner, hier in Mbasic für APPLE II:

```
10 DATA ws.com                              [572]
19 DATA ende                                [510]
20 REM Empfaengerprogramm 7-bit Centronics  [2462]
30 portb=&F8FD:portc=&F8FE:ports=&F8FF:REA  [3874]
D titel$
40 IF titel$="ende" THEN END                [1440]
50 DEFINT a-m:MODE 2                        [707]
60 OPENOUT "zahlen.zei"                     [999]
70 OUT ports,134 :'Initialisieren des Port  [4348]
s auf Mode 1
80 '                                        [117]
90 REM hier Einsprungpunkt der Schleife     [3204]
100 DIM a(15000)                            [372]
110 REM Warteschleife zum Aufsetzen         [2876]
120    IF INP(portc) AND 2 THEN 130 ELSE 12 [3200]
0
130    dummy=INP(portb) OR 128 :IF dummy=18 [3659]
0 THEN 120
140 IF dummy=191 THEN GOTO 280              [2289]
150 ' Einlesen des Blocks                   [2304]
160 a(1)=dummy:FOR i=2 TO 15000             [2481]
170    IF INP(portc) AND 2 THEN 180 ELSE 17 [2376]
0
180    a=INP(portb) OR 128:a(i)=a:IF a(i)=1 [2924]
81 THEN 210
190 NEXT                                    [350]
200 'Ausscheiden der redundanten Bytes      [5983]
210 CLS:IF a(i)=181 THEN a(i)=180:IF a(i-1  [4341]
)=181 THEN a(i-1)=180
220 ende=i:altbyte=a(1) :k=2 :PRINT#9,CHR$  [4492]
(a(1));
230 FOR j=2 TO ende                         [848]
240    IF a(j)=altbyte THEN GOTO 260        [1797]
250 PRINT#9,CHR$(a(j));:altbyte=a(j):k=k+1  [5814]
: IF a(j)=180 THEN PRINT#9,CHR$(13);
260 NEXT j                                  [466]
270 PRINT "bitte jetzt weiter":GOTO 120     [2737]
280 PRINT#9,CHR$(191);:ERASE a              [1842]
290 CLOSEOUT:CLOSEIN:OPENIN "zahlen.zei":O  [5848]
PENOUT titel$:WIDTH 255:endeflag=0
300 blockzaehler=0:DIM b(1000)              [2327]
310 DEF FNhnibble$(x)=LEFT$(HEX$(x,2),1)    [1728]
```

```
320 DEF FNlnibble$(x)=RIGHT$(HEX$(x,2),1):  [3208]
DEF FNwert(x$)=VAL("&"+x$)
330 blockzaehler=blockzaehler+1:DIM wert$(  [5045]
128):FOR i=1 TO 128:wert$(i)="??" :NEXT i
340 i=1                                     [545]
350 IF EOF THEN 420                         [78]
360 LINE INPUT#9,dummy$                     [2161]
370 FOR j=1 TO LEN(dummy$)                  [1490]
380 b(i)=ASC(MID$(dummy$,j,1)):IF b(i)=180  [3954]
 THEN 420
390 IF b(i)=191 THEN endeflag=1             [1479]
400 i=i+1:NEXT j                            [688]
410 GOTO 350                                [470]
420 ende=i :i=0                             [1242]
430 i=i+1                                   [307]
440 IF b(i)=180 THEN GOTO 530               [1231]
450 IF FNhnibble$(b(i))="A" THEN offset=(F  [5948]
Nwert(FNlnibble$(b(i)))*32)-32:GOTO 460
 ELSE 430
460 '                                       [117]
470 i=i+1:whn$=FNhnibble$(b(i))             [1926]
480 IF whn$="B" THEN GOTO 440               [2530]
490 IF whn$="C" OR whn$="D" THEN 500 ELSE   [3136]
470
500 adr=offset+(FNwert(FNlnibble$(b(i))))+  [3385]
((b(i) AND 16)+1)
510 IF FNhnibble$(b(i+1))="8" AND FNhnibbl  [6790]
e$(b(i+2))="9" THEN wert$(adr)=FNlnibble$(
b(i+1))+FNlnibble$(b(i+2)):i=i+2 :GOTO 470
```

```
520 GOTO 470                                [454]
530 REM hier einsprung fuer Auswertung      [2487]
540 FOR i=1 TO 128                          [823]
550 IF INSTR(1,wert$(i),"?")>0 THEN 560 EL  [4196]
SE PRINT#9,CHR$(FNwert(wert$(i)));:GOTO 66
0
560 IF endeflag=1 THEN 720                  [1584]
570 PRINT "Im Block ";blockzaehler;" : "    [3096]
580 PRINT:PRINT TAB(1);"Position ";TAB(20)  [3347]
;"Wert"
590 IF i<4 THEN GOTO 630                    [1118]
600 PRINT TAB(1);i-3;TAB(20);Wert$(i-3)     [2299]
610 PRINT TAB(1);i-2;TAB(20);Wert$(i-2)     [2510]
620 PRINT TAB(1);i-1;TAB(20);Wert$(i-1)     [2260]
630 PRINT TAB(1);i;TAB(20);Wert$(i)         [1667]
640 INPUT "Neuer Wert ";wert$(i)            [2327]
650 GOTO 550                                [431]
660 NEXT i                                  [471]
670 PRINT STRING$(80,32):PRINT "Der Block   [7254]
";blockzaehler;" : ":PRINT:FOR i=1 TO 128
680 IF FNwert(wert$(i))<32 OR FNwert(wert$  [4045]
(i))>128 THEN a$="." ELSE a$=CHR$(FNwert(w
ert$(i)))
690 PRINT a$;:NEXT                          [765]
700 ERASE wert$                             [553]
710 GOTO 330                                [506]
720 CLOSEOUT:CLOSEIN:endeflag=0 :|ERA,"b:*  [4272]
.zei":ERASE b,wert$:GOTO 30
```

Das Programm muß eine hohe Datensicherheit gewährleisten, da ein falsches Byte in einer Maschinencode-Datei ein ganzes Programm wertlos machen kann. Dies wird durch zwei Dinge gewährleistet. Zum einen erkennt das Empfangsprogramm kleine Übertragungsfehler durch den Aufbau des Protokolls selbst, zum anderen wird jedes Informationsbyte zweimal übertragen.

Das Empfangsprogramm

Die Aufnahme der Zeichen erfolgt wie in dem Programm ASCII-EMP., das Bit 8 wird durch eine OR-Verknüpfung permanent auf logisch 1 gesetzt. Sobald das erste Zeichen, das ungleich 180 (= Synchronisationsbyte und Anfangskennzeichen) ist, eintrifft, geht der CPC in eine Aufnahmeschleife, die erst nach Aufnahme des Zeichens 181 (= Ende einer 32-Blockgruppe) wieder verlassen wird.

Danach wird das Array (Variable a%) auf redundante Bytes geprüft und dann weggeschrieben. Jetzt geht der Rechner wieder in die Synchronisationsschleife.

Trifft hier das Ende-Kennzeichen ein, so wird die Datei "zahlen.zei" ausgewertet.

Hierfür wird eine Datei auf den Namen der zu erzeugenden Datei angelegt (Ausgabedatei), als Eingabedatei fungiert "zahlen.zei".

Die Daten werden in einer Schleife eingelesen und ausgewertet, deren Abbruchbedingung das Ende-Kennzeichen (EOF) der Eingabedatei ist.

Die Datenbytes werden in ein High- und ein Low-Nibble zerlegt, das High-Nibble enthält die Steuerinformation nach dem Protokoll, so daß eine eindeutige Zuordnung möglich ist.

Bei Übertragungsfehlern fragt das Programm nach dem richtigen Wert unter Angabe des Übertragungsblocks und der Adresse im Block. Den richtigen Wert kann man dann auf dem Host-Rechner mit Hilfe von DDT.COM feststellen. Nach der Auswertung wird die Datei "zahlen.zei" gelöscht und die nächste Datei kann übertragen werden.

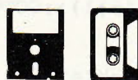
H. Jakstatt

Das Empfangsprogramm für den Schneider CPC:

```
3 REM Programm PORTPRO.SEN                  [2998]
4 DATA WS.COM,136,1                         [1010]
20 DATA ENDE,0,0                            [358]
30 HOME                                     [306]
40 READ TITEL$,ZAHL,ZAHL1                   [2764]
50 PRINT:PRINT TITEL$ ;" VON";ZAHL1;"BIS"   [4302]
;ZAHL
60 IF ZAHL=0 THEN END                       [1135]
70 ZEIT=1 : WIDTH 255:ENDE=28000            [1382]
80 OPEN "R",1,TITEL$                        [305]
90 FIELD 1,128 AS RECORD$                   [982]
100 ANFANG=ZAHL1                            [641]
110 ZAHLEND=ZAHL1+31                        [1156]
120 IF ZAHLEND>= ZAHL THEN ZAHLEND=ZAHL :   [1986]
ENDEFLAG=1
130 FOR Q=ZAHL1 TO ZAHLEND                  [1804]
140 GET 1,Q                                 [172]
150 ZEICHEN=180:GOSUB 420:GOSUB 420:GOSUB   [2614]
420
160 FOR K=0 TO 3                            [1299]
170 IF K=0 AND Q=ANFANG THEN ZEIT=100 ELSE  [1881]
 ZEIT=1
180 OFFSET=K*32                             [940]
190 ZEICHEN=(K+1) OR 160:GOSUB 420          [1342]
200 FOR J=1 TO 32                           [738]
210 BYTE=ASC(MID$(RECORD$,J+OFFSET,1))      [816]
220 HEXBYTE$=RIGHT$("00"+HEX$(BYTE),2)      [1693]
230 LBYTE$=LEFT$(HEXBYTE$,1)                [1374]
```

```
240 HBYTE$=RIGHT$(HEXBYTE$,1)               [1844]
250 ADR=(J-1)                               [558]
260 ZEICHEN=ADR OR 192:GOSUB 420            [1627]
270 ZEICHEN=VAL("&H"+LBYTE$) OR &80:GOSUB   [2852]
420
280 ZEICHEN=VAL("&H"+HBYTE$) OR &90:GOSUB   [3311]
420
290 NEXT J                                  [370]
300 ZEICHEN=(K+1) OR 176:GOSUB 420          [2093]
310 NEXT K                                  [373]
320 PRINT Q;                                [560]
330 A=FRE("")                               [1439]
340 NEXT Q                                  [367]
350 ZEICHEN=181 : GOSUB 420                 [1717]
360 ZAHL1=ZAHL1+32                          [1111]
370 FOR I=1 TO ENDE : IF INKEY$<>"" THEN 3  [3948]
80 ELSE NEXT I
380 ENDE=28000                              [235]
390 IF ENDEFLAG=1 THEN GOTO 400 ELSE GOTO   [1737]
110
400 ZEICHEN=191 : FOR I=1 TO 10 : GOSUB 42  [2724]
0 : NEXT I
410 GOTO 430                                [462]
420 FOR R=1 TO 2: LPRINT CHR$(ZEICHEN);:FO  [3462]
R S=1 TO ZEIT:NEXT S:NEXT R:RETURN
430 ENDEFLAG=0 : BLOCK=ZAHL-ANFANG          [2021]
440 BLOCK1=INT(BLOCK/16)+1                  [873]
450 CLOSE                                   [290]
460 FOR I=1 TO (BLOCK1*100000 !):NEXT I     [2683]
```

BACKLIST



für 464-664-6128

Wer hat noch nicht die Erfahrung gemacht, daß ein Programm auf Diskette schneller gelöscht ist, als man denkt?

Nun geht die Suche nach einer anderen Kopie dieses Programms los, hat man sie, ist es gut, wenn nicht, dann Ade schönes Programm.

Um so eine Situation zu umgehen, gibt es zwei Möglichkeiten:

1. Von jedem Programm auf Diskette eine Backupkopie auf eine zweite Diskette ablegen.
   (Preis pro Diskette ca. 15,– DM.)

2. Von jeder Diskette eine Backupkopie auf Kassette erstellen.
   (Preis pro Kassette ca. 3,50 DM.)

Während der Vorteil der Diskette als Backup-Speichermedium in der Schnelligkeit und Datensicherheit liegt, ist er bei der Kassette im Preis. So dauert eine Kopie einer ganzen Diskettenseite bei 2800 Baud ca. 12 Minuten (möglich sind 3600, wobei jedoch nur absolut hochwertiges Bandmaterial verwendet werden sollte). Dies fällt jedoch kaum ins Gewicht, da das Programm selbständig zwischen Tape und Disc umschaltet und somit unbeaufsichtigt laufen kann.
Voraussetzung dafür ist ein Kassettenrecorder mit Kopfhörer- bzw. Lautsprecherausgang, sowie einem Mikrofoneingang. Der Kopfhörerausgang wird mit den Pins 2 und 4 der Kassettenanschlußbuchse an der Rückseite Ihres Computers verbunden; der Mikrofoneingang mit den Pins 2 und 5. Wenn keine externe Motorsteuerung für den Kassettenrecorder vorgesehen ist, kann man sich wie folgt behelfen:

- Kassettenrecorder aufschrauben

- eine der Zuleitungen vom Kassettenrecorder zum Kassettenmotor auftrennen

- die beiden Enden der durchtrennten Leitung mit den Pins 1 und 3 der Kassettenanschlußbuchse verbinden.

Die Anschlußbelegung und Pin-Nummern der Kassettenanschlußbuchse finden Sie im Handbuch, Kapitel 7, Seite 37.

Nun zum eigentlichen Programm:
Nachdem das Programm mit RUN gestartet wurde, erscheint auf dem Bildschirm das Hauptmenue. Der Bildschirm ist in zwei Bereiche gegliedert (immer):
Ein Ausgabebereich (für Catalog usw.) und ein Bereich, in dem der Computer dem Benutzer Anweisungen gibt.
Vom Hauptmenue aus sind folgende Punkte auswählbar:

(1) Diskette auf Kassette abspeichern
Der Catalog der Diskette und ihr Format wird auf dem Bildschirm ausgegeben, dann fragt der Computer, ob die Diskette wirklich auf Kassette abgespeichert werden soll. Darauf antwortet man entweder mit Ja (="J") oder Nein (="N"). Bei "N" findet man sich im Hauptmenue wieder, bei "J" wird zuerst der Catalog (was zur späteren Identifizierung der Kassette dient) und dann die gesamte Diskette auf Band geschrieben.

(2) Kassette auf Diskette zurückschreiben
Der Catalog wird eingeladen, dann wird abgefragt, ob man die Kassette wirklich auf Diskette zurückschreiben will. Bei "N" geht es zurück ins Hauptmenue, bei "J" wird die Kassette auf Diskette zurückgeschrieben.

(3) Kassettenmotorsteuerung
Hier kann man den Kassettenmotor ein- bzw. ausschalten, ohne daß Daten gelesen bzw. geschrieben werden. Hierbei bedeutet "A" gleich Motor an, "B" gleich Motor aus und "C" gleich zurück ins Hauptmenue.

(4) Baudrate für Speichern festlegen
Hier kann die Baudrate zum Abspeichern auf Diskette zwischen 800 und 3600 Baud gewählt werden. Im Hauptmenue wird die aktuelle Baudrate angezeigt.

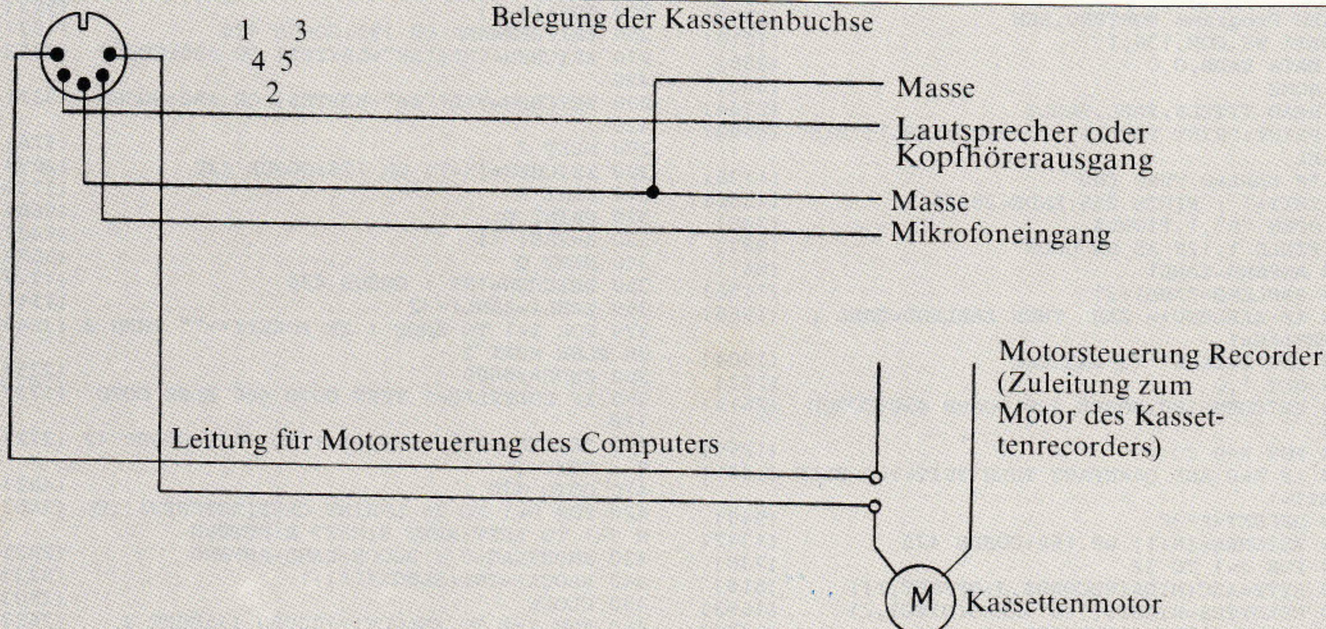
(5) Diskette catalogisieren
Entspricht dem CAT-Kommando im Basic, wobei jedoch zusätzlich noch das Diskettenformat angezeigt wird.

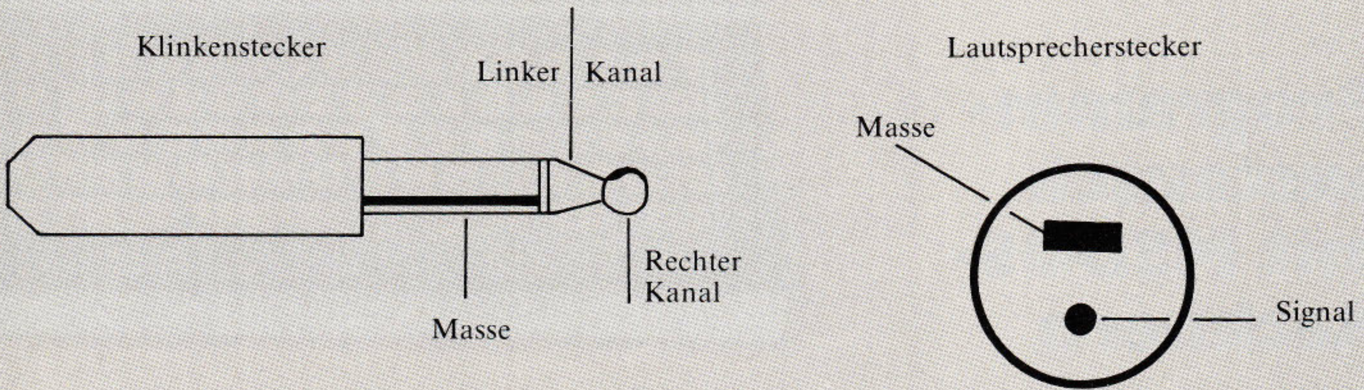
(6) Programm beenden
Beendet das Programm

Wenn eine Schreib- bzw. Leseoperation von/auf Diskette ausgeführt werden soll, jedoch keine Diskette im Laufwerk A liegt, erscheint die Fehlermeldung "Keine Diskette in A, bitte einlegen, dann (ENTER)", wenn die Diskette beschrieben werden soll. Ist aber die Diskette schreibgeschützt, erscheint die Fehlermeldung "DISKETTE ist schreibgeschützt, Schutzzunge verschieben, dann (ENTER)".
Im ersten Fall sollte man eine Diskette einlegen, dann die (ENTER)-Taste betätigen. Im zweiten Fall die Diskette entnehmen, die rote Schutzzunge am oberen Diskettenrand so verschieben, bis ein roter Punkt links oben auf der zu beschreibenden Diskettenseite erscheint.

Sollte beim Einlesen der Kassette ein READ ERROR auftreten, so muß man die Diskette noch einmal, jedoch mit kleinerer Baudrate abspeichern. Sollte das immer noch nichts nützen, so ist zu empfehlen, qualitativ bessere Kassetten zu verwenden.

Peter Rühl






```
10 ' ************************************** [9251]
**************************************
 *              DISC-TAPE-BACKUP   KEY 1986
  by  Peter Ruehl                  *    *
**************************************
******************************
20 MEMORY 12999:MODE 2:WINDOW#1,1,80,1,22: [8579]
WINDOW#2,1,80,23,25:BORDER 1:INK 0,1:INK 1
,24:PAPER#1,0:PEN#1,1:PAPER#2,1:PEN#2,0:WI
NDOW SWAP 0,1:CLS#2:CLS
30 SPEED WRITE 0:kass=1000:catalog=&9000:d [16299]
isc=&903A:befehl=&9030:ram=&9035:track=&90
33:sector=&9034:offset=&9032:buffer=13002:
lowbyte=&9066:highbyte=&9067:cspeed=&9065:
cwork=&9077:tape=&906E:GOSUB 10000
1000 '                                     [117]
1010 ' *****  HAUPTMENUE  *****            [948]
1020 '                                     [117]
1030 CLS:CLS#2:PRINT CHR$(150);STRING$(78, [10776]
154);CHR$(156);CHR$(149);STRING$(27,32);"*
 DISC-TAPE-BACKUP *";STRING$(31,32);CHR$(1
49);CHR$(149);
1040 PRINT"     Version 1.0           ==== [8215]
================              ";CHR$(164);" 1986
 by P. Ruehl      ";CHR$(149);CHR$(147);STR
ING$(78,154);CHR$(153)
1050 LOCATE 18,7:PRINT"[1]  Diskette auf K [18616]
assette abspeichern":LOCATE 18,9:PRINT"[2]
  Kassette auf Diskette zurueckschreiben":
LOCATE 18,11:PRINT"[3]  Kassettenmotorsteu
erung":LOCATE 18,13:PRINT"[4]  Baudrate fu
er Speichern festlegen"
1060 LOCATE 18,15:PRINT"[5]  Diskette Cata [16379]
logisieren":LOCATE 18,17:PRINT"[6]  Progra
mm beenden":LOCATE 11,20:PRINT"Die Eingest
'ellte Schreibgeschwindigkeit betraegt ";ka
ss;" Baud.":LOCATE#2,21,2:PRINT#2,"Druecke
n Sie die gewuenschte Zahl"
1070 abfr$=INKEY$:IF abfr$="" THEN 1070 EL [4682]
SE IF ASC(abfr$)<49 OR ASC(abfr$)>54 THEN
1070
1080 abfr=(ASC(abfr$))-48:ON abfr GOTO 200 [4659]
0,3000,5000,4000,9000,8000
1090 GOTO 1070                             [317]
2000 '                                     [117]
2010 ' *****  CATALOG ABSPEICHERN  *****   [2071]
2020 '                                     [117]
2030 CLS:CLS#2:GOSUB 11000:IF nodisc=0 THE [2803]
N GOSUB 12000
2040 |DISC.IN:|TAPE.OUT:CAT:format=PEEK(&A [12525]
89F)-1:IF format=0 THEN PRINT"IBM - Format
" ELSE IF format=&C0 THEN PRINT"DATA-ONLY
- Format" ELSE IF format=&40 THEN PRINT"CP
/M - Format" ELSE PRINT"RANDOM - Format"
2050 LOCATE#2,16,2:PRINT#2,"Wirklich auf K [5141]
ASSETTE schreiben (J)a oder (N)ein ?"
2060 abfr$=UPPER$(INKEY$):IF abfr$<>"J" AN [6088]
D abfr$<>"N" OR abfr$="" THEN 2060
2070 IF abfr$="N" THEN 1000                [1472]
2080 CLS#2:LOCATE#2,21,2:PRINT#2,"Druecken [7322]
 Sie REC und PLAY, dann [ENTER]"
2090 WHILE INKEY$<>CHR$(13) : WEND         [1281]
2100 CLS#2:CALL catalog:LOCATE#2,27,2:PRIN [5129]
T#2,"CATALOG wird abgespeichert"
2110 bytes=PEEK(13000)+256*PEEK(13001)     [2591]
2120 OPENOUT"!catalog"                     [1233]
2130 PRINT#9,format                        [1534]
2140 FOR x=buffer TO bytes                 [2180]
2150 WRITE#9,CHR$(PEEK(x))                 [1556]
2160 NEXT x                                [452]
2170 CLOSEOUT                              [902]
2180 CLS#2                                 [372]
2190 GOTO 6000                             [395]
3000 '                                     [117]
3010 ' *****  CATALOG EINLADEN  *****      [1805]
3020 '                                     [117]
3030 |DISC.OUT:|TAPE.IN:CLS:CLS#2:LOCATE#2 [6744]
,25,2:PRINT#2,"Druecken Sie PLAY, dann [EN
TER]"
3040 WHILE INKEY$<>CHR$(13) : WEND         [1281]
3050 CLS#2:LOCATE#2,30,2:PRINT#2,"CATALOG  [3286]
wird geladen":LOCATE 1,1
3060 OPENIN"!catalog"                      [2286]
3070 INPUT#9,format                        [1528]
3080 ram1=buffer-1                         [1231]
3090 WHILE NOT EOF                         [1840]
3100 ram1=ram1+1:INPUT#9,a$:POKE ram1,ASC( [2585]
a$)
3110 WEND                                  [390]
3120 CLOSEIN                               [752]
3130 FOR x=buffer TO ram1:PRINT CHR$(PEEK( [3969]
x));:NEXT
3140 CLS#2:LOCATE#2,12,2:PRINT#2,"Wirklich [6159]
 auf DISKETTE zurueckschreiben (J)a oder (
N)ein ?"
3150 abfr$=UPPER$(INKEY$):IF abfr$<>"J" AN [6007]
D abfr$<>"N" OR abfr$="" THEN 3150
3160 IF abfr$="N" THEN 1000 ELSE 7000      [1970]
4000 '                                     [117]
4010 ' *****  BAUDRATE FESTLEGEN  *****    [1007]
4020 '                                     [117]
4030 CLS:CLS#2                             [239]
4040 LOCATE 18,2:PRINT"Folgende Baudraten  [4621]
sind moeglich:"
4050 PRINT:PRINT                           [743]
4060 FOR x=1 TO 8                          [702]
4070 PRINT"                 [";x;"]   ---  [6658]
------>  ";:IF x=1 THEN PRINT" ";(x+1)*400
;" Baud" ELSE PRINT (x+1)*400;" Baud"
4080 PRINT                                 [361]
4090 NEXT x                                [452]
4100 LOCATE#2,5,2:PRINT#2,"Bitte druecken  [8337]
Sie die Ziffer, die vor der gewuenschten B
audrate steht."
4110 abfr$=INKEY$:IF abfr$="" THEN 4110    [1361]
4120 abfr=ASC(abfr$)                       [1144]
4130 IF abfr<49 OR abfr>56 THEN 4110       [2457]
4140 zwi=INT(1000000/((abfr-47)*1200)+0.5) [3246]
4150 hbyte=INT(zwi/256)                    [2194]
4160 lbyte=zwi-hbyte*256                   [2064]
4170 POKE lowbyte,lbyte:POKE highbyte,hbyt [3573]
e
4180 CALL cspeed                           [615]
4190 kass=(abfr-47)*400                    [1400]
4200 GOTO 1000                             [339]
5000 '                                     [117]
5010 ' *****  KASSETTENMOTORSTEUERUNG  *** [2003]
**
5020 '                                     [117]
5030 CLS:CLS#2:LOCATE 20,7:PRINT"[A]  Kass [18093]
ettenmotor einschalten":LOCATE 20,10:PRINT
"[B]  Kassettenmotor ausschalten":LOCATE 2
0,13:PRINT"[C]  Zurueck zum Hauptmenue":LO
CATE#2,22,2:PRINT#2,"Druecken Sie die gewu
enschte Taste"
5040 LOCATE 18,18:PRINT"Kassettenmotor Sta [4359]
"
5050 abfr$=UPPER$(INKEY$):IF abfr$="" OR a [5924]
bfr$<>"A" AND abfr$<>"B" AND abfr$<>"C" TH
EN 5050
5060 abfr=(ASC(abfr$))-64:ON abfr GOTO 508 [3209]
0,5100,1030
5070 GOTO 5050                             [337]
5080 '                                     [117]
5090 LOCATE 18,18:PRINT"Kassettenmotor Sta [5016]
  --  <AUS>":CALL &BC6E:GOT
O 5050
5100 '                                     [117]
5110 LOCATE 18,18:PRINT"Kassettenmotor Sta [6599]
":CALL &BC71:GOTO 5050

6000 '                                     [117]
6010 ' *****  DISKETTE AUF BAND SICHERN  * [1842]
****
6020 '                                     [117]
6030 GOSUB 11000:IF nodisc=0 THEN GOSUB 12 [3387]
000
6040 POKE befehl,&84:POKE offset,format:PO [2850]
KE cwork,&9E
6045 FOR x=0 TO 7                          [695]
6050 buffer1=buffer                        [2308]
6060 FOR y=0 TO 4                          [1454]
6070 FOR z=1 TO 9                          [1282]
6080 POKE sector,z:POKE track,x*5+y:POKE r [9576]
am,buffer1-(INT(buffer1/256)*256):POKE ram
+1,INT(buffer1/256):CALL disc
6090 buffer1=buffer1+512                   [2526]
6100 NEXT z                                [450]
6110 LOCATE#2,31,2:PRINT#2,"TRACK ";x*5+y; [2549]
" wird kopiert"
6120 NEXT y                                [455]
6130 CALL tape                             [631]
6140 NEXT x                                [452]
6150 CLS#2:LOCATE#2,15,2:PRINT#2,"DISKETTE [6222]
 komplett abgespeichert, weiter mit [ENTER
]"
6160 WHILE INKEY$<>CHR$(13) : WEND:GOTO 10 [1856]
00
7000 '                                     [117]
7010 ' *****  BAND AUF DISKETTE ZURUECKSCH [2022]
REIBEN  *****
7020 '                                     [117]
7030 CLS#2:GOSUB 11000:IF nodisc=0 THEN GO [5274]
SUB 12000 ELSE IF wrprot=64 THEN GOSUB 130
00
7040 LOCATE#2,18,2:PRINT#2,"KASSETTE wird  [8357]
auf DISKETTE zurueckgeschrieben":POKE befe
hl,&85:POKE offset,format:POKE cwork,&A1
7050 FOR x=0 TO 7                          [695]
7060 buffer1=buffer                        [2308]
7070 CALL tape                             [631]
7080 FOR y=0 TO 4                          [1454]
7090 FOR z=1 TO 9                          [1282]
7100 POKE sector,z:POKE track,x*5+y:POKE r [9576]
am,buffer1-(INT(buffer1/256)*256):POKE ram
+1,INT(buffer1/256):CALL disc
7110 buffer1=buffer1+512                   [2526]
7120 NEXT z                                [450]
7130 CLS#2:LOCATE#2,31,2:PRINT#2,"TRACK "; [4662]
x*5+y;" wird geladen"
7140 NEXT y                                [455]
7150 NEXT x                                [452]
7160 CLS#2:LOCATE#2,18,2:PRINT#2,"KASSETTE [6628]
 komplett geladen, weiter mit [ENTER]"
7170 WHILE INKEY$<>CHR$(13) : WEND:GOTO 10 [1856]
00
8000 '                                     [117]
8010 ' *****  PROGRAMM BEENDEN  *****      [1346]
8020 '                                     [117]
8030 MODE 2:|DISC:END                      [1583]
9000 '                                     [117]
9010 ' *****  DISKETTE CATALOGISIEREN  *** [1765]
**
9020 '                                     [117]
9030 CLS:CLS#2:GOSUB 11000:IF nodisc=0 THE [2803]
N GOSUB 12000
9040 CAT:format=PEEK(&A89F)-1              [694]
9050 IF format=0 THEN PRINT"IBM - Format"  [9183]
ELSE IF format=&C0 THEN PRINT"DATA-ONLY -
Format" ELSE IF format=&40 THEN PRINT"CP/M
 - Format" ELSE PRINT"RANDOM - Format"
9060 LOCATE#2,23,2:PRINT#2,"Druecken Sie [ [4217]
ENTER] fuer Hauptmenue"
9070 WHILE INKEY$<>CHR$(13) : WEND:GOTO 10 [1856]
00
10000 '                                    [117]
10010 ' *****  DATAS FUER MASCHINENPROGRAM [3821]
ME EINPOKEN  *****
10020 '                                    [117]
10030 DATA &21,&01,&01,&CD,&75,&BB,&21,&CA [2038]
,&32,&E5,&CD,&60,&BB,&E1,&77,&23, 1925
10040 DATA &FE,&74,&28,&14,&E5,&CD,&78,&BB [4105]
,&24,&7C,&FE,&51,&28,&05,&CD,&75, 2033
10050 DATA &BB,&18,&E7,&26,&01,&2C,&18,&F6 [3714]
,&2B,&22,&C8,&32,&C9,&00,&00,&00, 1323
10060 DATA &00,&00,&00,&00,&00,&00,&00,&00 [2530]
,&00,&00,&3A,&30,&90,&47,&21,&63, 453
10070 DATA &90,&70,&CD,&D4,&BC,&D0,&22,&60 [2895]
,&90,&79,&32,&62,&90,&21,&32,&90, 1983
10080 DATA &7E,&23,&56,&23,&4E,&81,&4F,&1E [4523]
,&00,&2A,&35,&90,&DF,&60,&90,&C9, 1501
10090 DATA &00,&00,&00,&00,&00,&21,&5D,&00 [2937]
,&3E,&0A,&CD,&68,&BC,&C9,&3E,&16, 980
10100 DATA &11,&00,&5A,&21,&CA,&32,&CD,&9E [2614]
,&BC,&C9, 1144
10110 dat=0 : sz=0 : dz = 10030            [1776]
10120 FOR adr =-28672  TO -28551           [1468]
10130   READ byte : dat=dat+1              [1921]
10140   sz=sz+byte                         [1743]
10150   POKE adr,byte                      [898]
10160   IF dat < 16 AND adr < -28551 THEN  [2049]
10200
10170   READ chksum                        [1079]
10180   IF chksum<>sz THEN PRINT "Fehler i [3061]
n Zeile :";dz
10190   dz=dz + 10 : sz=0 : dat=0          [2078]
10200 NEXT adr                             [453]
10210 RETURN                               [555]
11000 '                                    [117]
11010 ' *****  STATUS DISKETTENLAUFWERK  * [3179]
****
11020 '                                    [117]
11030 OUT(&FA7E),1:FOR x=1 TO 1000:NEXT:OU [6237]
T(&FB7F),4:OUT(&FB7F),(-PEEK(&A700)+2):sta
tus=INP(&FB7F):OUT(&FA7E),0
11040 nodisc=status AND 32:wrprot=status A [3525]
ND 64
11050 RETURN                               [555]
12000 '                                    [117]
12010 ' *****  KEINE DISKETTE IN LAUFWERK  [2342]
A  *****
12020 '                                    [117]
12030 LOCATE#2,16,2:PRINT#2,"Keine DISKETT [4083]
E in A, bitte einlegen, dann [ENTER]"
12040 WHILE INKEY$<>CHR$(13) : WEND:CLS#2: [3142]
RETURN
13000 '                                    [117]
13010 ' *****  DISKETTE SCHREIBGESCHUETZT  [2950]
 *****
13020 '                                    [117]
13030 CLS#2:LOCATE#2,6,2:PRINT#2,"DISKETTE [10935]
 ist schreibgeschuetzt, Schutzzunge versch
ieben, dannn [ENTER]"
13040 WHILE INKEY$<>CHR$(13) : WEND:CLS#2: [3142]
RETURN
```

Programmaustausch mit MSDOS-Rechnern



Obwohl das Betriebssystem CP/M für 8-Bit-Rechner im Zeitalter der 16- und 32-Bit-Generation schon längst totgesagt war, erfreut es sich zur Zeit, nicht zuletzt auch durch die vielen Besitzer eines Schneider Computers, wieder großer Beliebtheit. Auch im Joyce schlägt einmal mehr das Herz des sehr bekannten Prozessors Z80 von Zilog.



Natürlich weiß jeder, daß es mittlerweile viel schnellere und komfortablere Microprozessoren gibt, die intern 16- oder sogar 32 Bit breite Datenworte verarbeiten und auch einen wesentlich größeren Speicherbereich adressieren können. Ihre höhere Verarbeitungsgeschwindigkeit resultiert aber nicht nur aus der Tatsache, daß über den um den Faktor 2 bzw. 4 vergrößerten Datenbus mehr Daten im Prozessor transportiert werden können. Auch der Befehlssatz dieser moderneren Prozessoren wurde erheblich erweitert, wodurch z.B. auch Multiplikationsbefehle verfügbar sind, so daß nicht jede arithmetische Operation auf eine simple Addition zurückgeführt werden muß. Diese Arbeit übernimmt jetzt der Prozessor.

Leider können jedoch in vielen Fällen die Möglichkeiten der Prozessoren nicht voll ausgeschöpft werden, da ihnen ein zu kleiner externer Datenbus verpaßt wurde. Auch der IBM PC und seine kompatiblen Brüder, die mit dem 16-Bit-Prozessor 8088 arbeiten, besitzen nur einen externen 8-Bit-Datenbus, wodurch Programme nur unwesentlich schneller abgearbeitet werden als beim Schneider. Damit wollen wir zum eigentlichen Thema dieses Artikels kommen:

Die drei Möglichkeiten

Um Programme zwischen MSDOS-Rechnern und dem Schneider CPC austauschen zu können, gibt es im Prinzip nur drei Möglichkeiten:

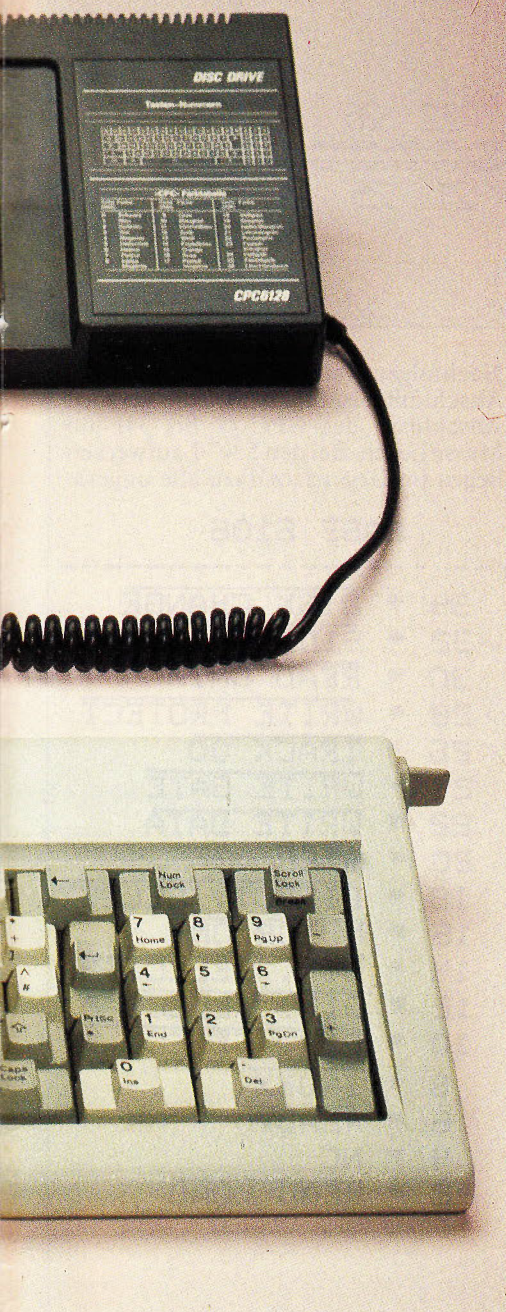
1. Eintippen
2. Austausch über eine serielle RS 232-Schnittstelle
3. Tansfer über das 5.25"-Laufwerk von Vortex

Bei allen drei Möglichkeiten ist natürlich klar, daß wir nur Source-Code, also Quell-Text im ASCII-Format transferieren können. Weiterhin müssen wir darauf achten, daß z.B. vor dem Ablauf eines Schneider-Basic-Programmes unter GWBASIC und MSDOS bzw. umgekehrt, die entsprechenden Syntax-Dialekte angepaßt werden müssen. Das geht aber ziemlich leicht, da sich die beiden Basics stark ähneln, was nicht zuletzt auch auf die gleiche Auflösung im Grafik-Modus zurückzuführen ist. WORDSTAR-Dateien sind sofort lauffähig, ebenso Source-Programme, die z.B. unter TURBO-Pascal geschrieben wurden, das ja ebenfalls unter MSDOS läuft.

Da das Eintippen sicher nicht näher erläutert werden muß, wollen wir uns jetzt mit den beiden letzten Punkten beschäftigen.

Transfer über die RS 232-Schnittstelle

Da der Schneider CPC leider keine serielle Schnittstelle besitzt, muß diese zunächst einmal erworben werden. Mittlerweile gibt es ja eine Vielzahl von Anbietern (s.a. Heft Nr. 12/85). Ein großer Nachteil ist jedoch, daß wir die beiden Rechner irgendwie koppeln müssen. Also entweder über das Postnetz, wobei wir zum zweiten Mal eine ganze Menge Aufwand treiben müssen (Modem etc.), oder indem wir unseren Schneider unter den Arm nehmen und



ihn neben dem IBM PC aufbauen. Danach wird das Verbindungskabel mit der vorhandenen RS 232-Schnittstelle des MSDOS-Rechners verbunden, die Parameter für das Übertragungsproprotokoll eingestellt und endlich kann es losgehen.

Transfer über eine 5.25"-Diskette

Viel schöner wäre es jedoch, wenn wir z.B. zu Hause mit unserem CPC einen längeren Text mit WORDSTAR unter CP/M erstellen würden, diesen auf eine 5.25"-Diskette speichern und irgendwo mit einem IBM PC wieder einlesen könnten, um ihn dann z.B. mit einem Schönschreib-Drucker ausdrucken zu lassen. Oder, wie schön wäre es, wenn wir Source-Texte in Turbo-Pascal, Basic oder FORTRAN von einem IBM-Rechner einfach per Diskette auf unseren Schneider übertragen könnten.

Um es gleich vorwegzunehmen, es geht. Alles, was man braucht, ist neben dem 5.25"-Laufwerk F1/S bzw. F1/D von Vortex, der Diskmanager PARA (s.a. Schneider International Nr. 7/85).

Nachdem das Programm MSDOS-MANAGER, mit dem man MSDOS-Diskettenformate direkt lesen und beschreiben konnte, leider wieder aus dem Softwareangebot von Vortex herausgenommen wurde, müssen wir uns also mit dem universellen PARA begnügen.

Arbeiten mit PARA

Nach dem Laden von PARA kann man sich getrost mit Hilfe der Menuesteuerung durch das Programm führen lassen. Nach dem Anwählen der Option 3 für spezielle Systeme könnemwir uns aus der von Vortex mitgelieferten, sehr reichhaltigen Systemdatei das Format IBM SS hereinladen und die Parameter mit der Option 2 installieren.

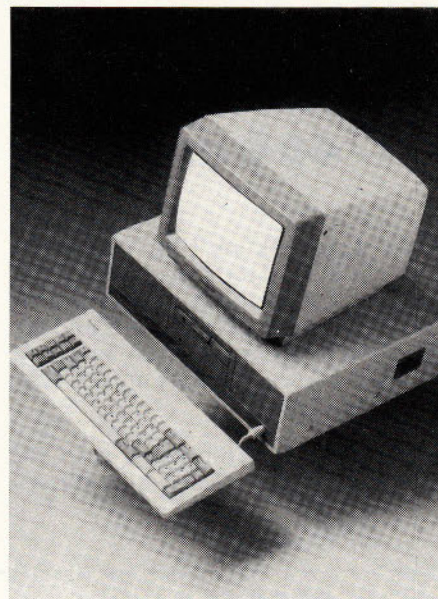
Wenn wir jetzt eine Diskette mit MS-DOS-Format einlegen, meldet PARA:

NO FILE

Das war allerdings auch zu erwarten, da nämlich die Directory unter MS-DOS ganz anders organisiert wird. Das installierte Format wird aber dennoch vom IBM-Rechner benutzt und zwar unter CP/M 86.

Arbeiten mit CP/M 86

Nicht nur MICROSOFT (MSDOS), sondern auch Digital Research, der Vater von CP/M, hat ein Betriebssystem für den 16-Bit-Mircroprozessor 8088 entwickelt. Dieses Betriebssystem heißt CP/M 86 und ist, wie der Name schon vermuten läßt, sehr stark an das uns bekannte CP/M 2.2 bzw. CP/M Plus angelehnt. Das ist wohl auch der Grund, weshalb IBM alle Rechner mit dem völlig neuen und moderneren MSDOS ausliefert. Der große Vorteil von CP/M 86 ist jedoch, daß man sofort mit ihm zurecht kommt, wenn man CP/M unter 8-Bit kennengelernt hat. Ein weiterer Vorteil von CP/M 86 besteht in der Tatsache, daß auch MSDOS-Programme erkannt und einwandfrei ausgeführt werden können, z.B. WORDSTAR. Wir können also sehr einfach Disketten unter CP/M 86 formatieren und beliebig beschreiben, z.B. mit einem Turbo-Pascal oder Basic-Source-Programm, das unter MSDOS erstellt wurde.



Ein kleines Problem gibt es noch bei WORDSTAR durch die unterschiedlichen Versionen. Da auf dem Schneider ohne Speichererweiterung normalerweise nur die Version 3.0 läuft, unter MSDOS aber häufig schon die neueren Versionen 3.3 und 3.4 anzutreffen sind, die ein paar nur unter 3.0 sichtbare Steuerzeichen in den Text einmischen, sollten alle Texte unter MSDOS nur über das ^P-Menue abgespeichert werden. Dabei müssen die Fragen "Für WORDSTAR?" unbedingt mit "n" und "Ausgabe auf Diskette?" mit "j" beantwortet werden.

Eine solche Diskette können wir nun einwandfrei mit PARA lesen und mit PIP auf eine Diskette im Vortex-Format kopieren. Besitzer des Einzellaufwerkes F1/S müssen den kleinen Umweg über die Kassette wählen, selbst wenn sie eine Vortex-Ramdisk ihr Eigen nennen können, da diese unter PARA in der jetzigen Version leider noch nicht angesprochen werden kann. Ein weiteres Problem taucht dadurch auch beim Beschreiben im IBM-Format auf, was bei der F1/S ebenfalls nur mit einem Trick behoben werden kann. Dazu benötigen wir nur ein einziges Mal das Doppellaufwerk F1/D und kopieren uns das Systemprogramm CASCOPY.COM auf die IBM-Arbeitsdiskette. Jetzt können wir auch mit dem Einzellaufwerk nach Herzenslust IBM-Disketten lesen und beschreiben.

Damit auch Besitzer des 3"-Laufwerkes von Schneider nicht leer ausgehen, können alle Programme, die mit CASCOPY und einer F1/S auf Kassette geschrieben wurden, mit dem Befehl CSAVE gelesen und auf eine 3"-Diskette gespeichert werden.

Was PARA sonst noch alles kann und wie es sich unter dem neuen Disketten-Betriebssystem VDOS 2.0 bewährt, erfahren Sie in einem weiteren Bericht.

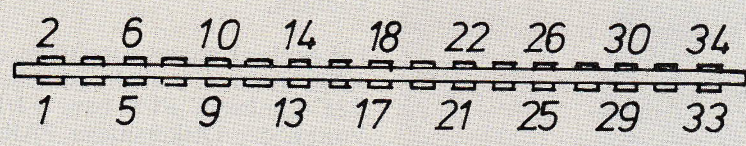
Dipl.Ing. H. Scheruhn

5 ¼"-Laufwerke am CPC 464 und 664

Das 3"-Laufwerk am CPC läßt kaum Wünsche offen. Trotzdem dürften 5 ¼"-Floppys als Alternative interessant sein -nicht nur vom Preis der Floppy und der Disketten her, sondern auch aufgrund der angebotenen Software unter CP/M auf 5 ¼"-Disketten.

Ein Blick in das Handbuch des CPC und ein Vergleich mit den Anschlüssen der Floppy zeigen, daß der Anschluß lediglich um 180 Grad verdreht ist. Skizze a) zeigt den Anschluß des CPC 664 und der Diskettenlaufwerke BASF und Tandon von der Rückseite.

Skizze a)

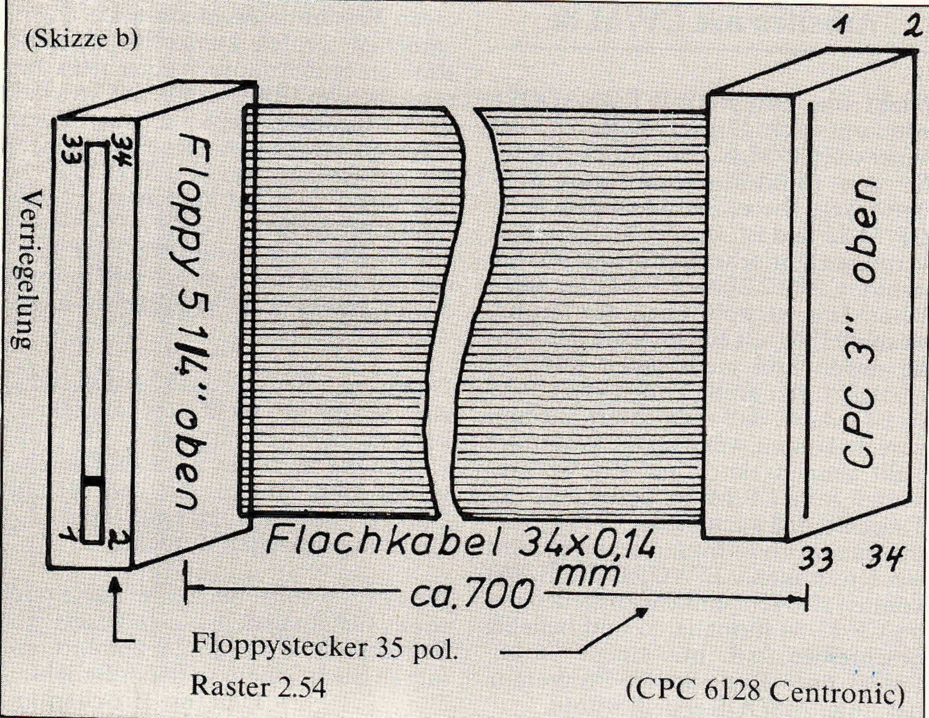


Nachfolgend eine Übersicht über die Anschlußbelegung, wobei die geraden Anschlüsse des CPC (2 bis 34) auf Masse liegen. Bei den 5 ¼"-Laufwerken liegen im Gegensatz dazu alle ungeraden, also Anschluß 1 bis 33, auf Masse. (Dies ist beim Anfertigen des Verbindungskabels unbedingt zu beachten (Skizze b))

CPC 664	BASF 6128 u. Tondon	BASF 6106
1 * READY	34 * READY	34 * DISK CHANGE
2 * SIDE SELECT	32 * SIDE SELECT	32 * SIDE SELECT
5 * READ DATA	30 * READ DATA	30 * READ DATA
7 * WRITE PROTECT	28 * WRITE PROTECT	28 * WRITE PROTECT
9 * TRACK 00	26 * TRACK 00	26 * TRACK 00
11 * WRITE GATE	24 * WRITE GATE	24 * WRITE GATE
13 * WRITE DATA	22 * WRITE DATA	22 * WRITE DATA
15 * STEP	20 * STEP	20 * STEP
17 * DIRECTION IN	18 * DIRECT. SELECT	18 * DIRECT. SELECT
19 * MOTOR ON	16 * MOTOR ON	16 * MOTOR ON
21 * NC	14 * DIR. SELECT 3	14 * DIR. SELECT 3
23 * DRIVE SELECT 1	12 * DIR. SELECT 2	12 * DIR. SELECT 2
25 * NC	10 * DIR. SELECT 1	10 * DIR. SELECT 1
27 * INDEX	8 * INDEX	8 * INDEX
29 * NC	6 * DIR. SELECT 4	6 * READY
31 * NC	4 * HEAD LOAD	4 * NC
33 * NC	2 * NC	2 * HEAD LOAD

(* NC = Nicht beschaltet)

(Skizze b)



Floppystecker 35 pol.
Raster 2.54

(CPC 6128 Centronic)

Man sollte bei der Auswahl der Floppy auf die Steprate achten. So läuft das BASF 6106 und 6128 ohne Änderung. Beim Tandonlaufwerk TM 50/1 und TM 50/2 muß die Steprate mit 'SETUP auf 15 msec gesetzt werden.

Anmerkung:
Das BASF 6128 und das Tandon 50/2 sind Doppellaufwerke, die zum Austesten genutzt wurden.
Bei diesen Laufwerken sind folgende Brücken zu schalten:

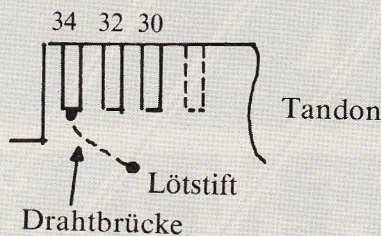
Tandon U 19 Pin 9 mit 8 und 14 mit 3, bei BASF 6128 AH und S2. Das »READY«-Signal ist bei Tandon nicht beschaltet (siehe dazu Skizze c).
Das READY-Signal beim BASF 6106 liegt auf Pin 6. Die Änderung der Leiterbahnführung ist auf Skizze d) zu sehen.

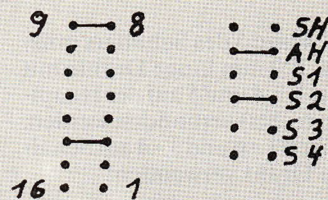
Für den CPC 464 wird zum Anschluß des Verbindungskabels noch ein Stück Leiterplatte, zweiseitig, 2 x 17 polig x 22 mm lang, benötigt. Falls im Handel nicht erhältlich, kann man sich dies aus einer handelsüblichen Leiterplatte (Eurokarte, Raster 2.54 x 2.54) zurechtschneiden.

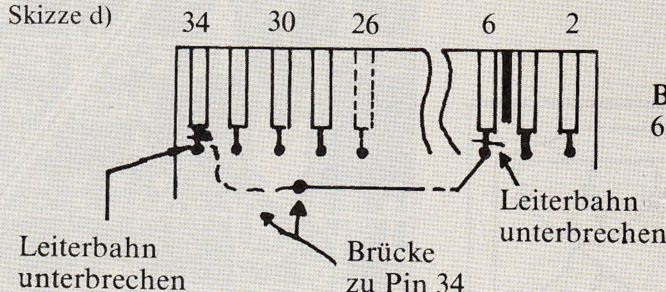
Beim Kauf der 34-poligen Floppystecker sollte man die Verriegelungsstücke mit erwerben und einstecken.
Das Formatieren mit dem CP/M-Programm FORMAT.COM ist nicht möglich. Bekannterweise wird unter COPYDISC eine Formatierung vorgenommen, wobei man anschließend die Files mit ERA löschen kann.
Beim CPC 6128 muß die Centronicsbelegung beachtet werden, wobei CPC-seitig jetzt eine Centronics-Buchse angeklemmt wird.

Die Anschlußbelegungen sind jeweils aus den technischen Unterlagen der Lieferfirmen entnommen.

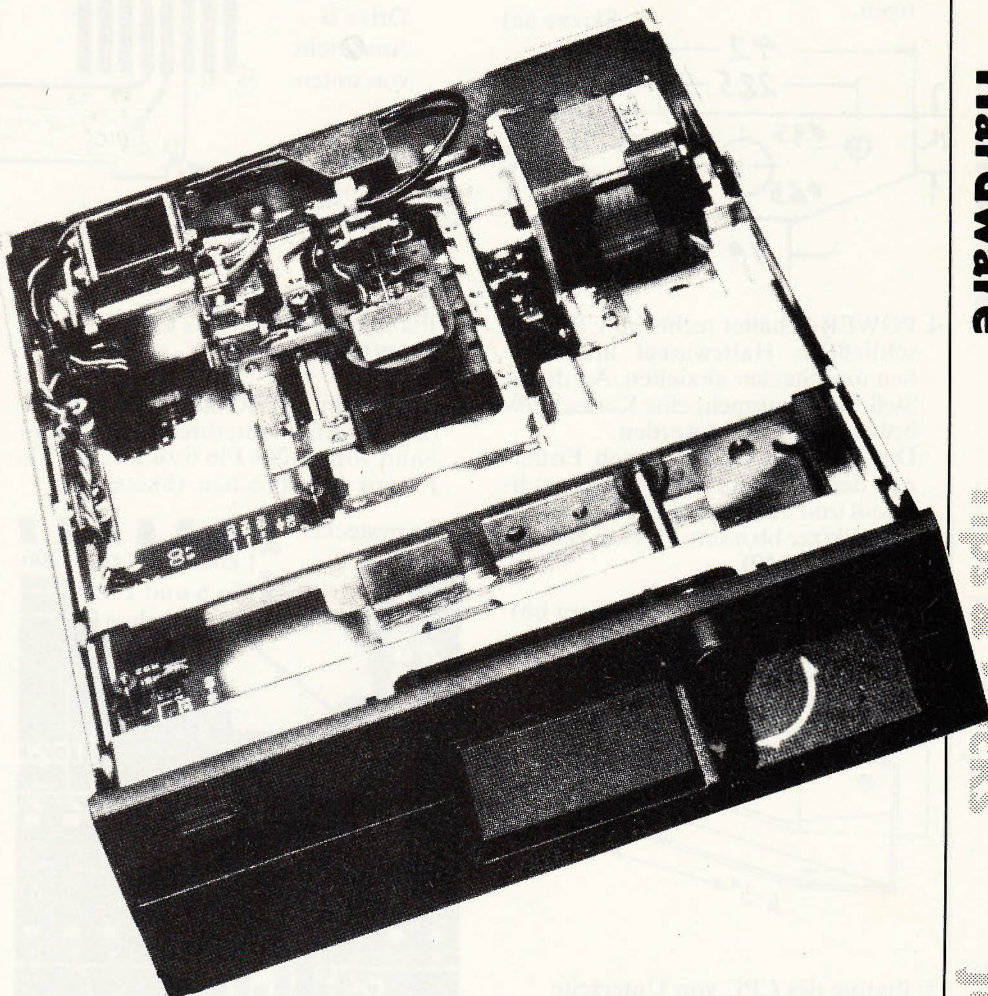
Hans Ziegler







5 ¼"-Laufwerke am CPC 664/6128 ohne Kompromisse



Mit dem nachfolgend beschriebenen Umbau ist man in der Lage, das 5 ¼"-Laufwerk als Laufwerk A, und die eingebaute 3"-Floppy als Laufwerk B, oder umgekehrt zu definieren.

Benötigt wird ein Micro- oder Schiebeschalter (2 x Um), ein Stück Alu- oder Stahlblech (13 x 42 mm) und vier flexible Drähte (ca. 120 mm lang).

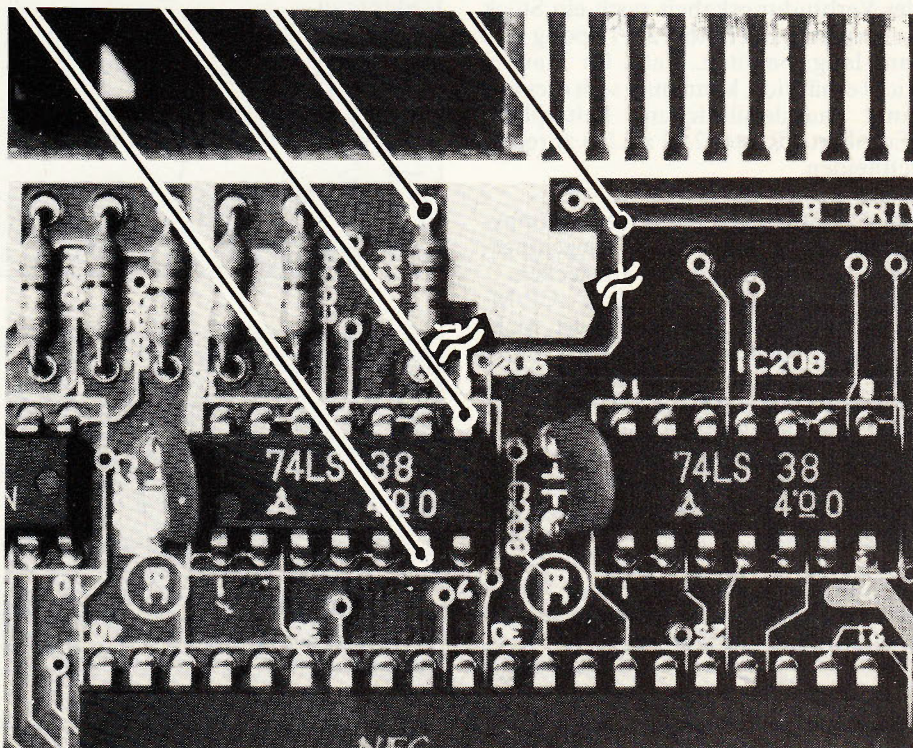
Bevor man nun mit dem Umbau beginnt, sollte man bedenken, daß bei einem solchen Eingriff der Garantieanspruch erlischt.

Das Verbindungskabel wird, wie unter Tip 5 ¼" am CPC 464/664 beschrieben, angefertigt.

Umbau:

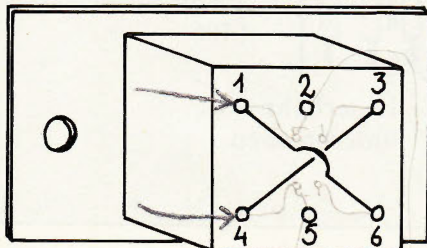
1. Alle Verbindungskabel abziehen, CPC umdrehen und die sieben Schrauben lösen.

2. CPC wieder in die normale Lage drehen und Oberteil abnehmen. Die Verbindung zur Tastatur lösen, indem man an der Halterung das Oberteil anhebt und das Band abzieht. Alle Steckverbindungen abziehen.



Die Pins 1, 2, 4 und 5 des Schalters müssen durch ca. 120 mm lange Drähte mit den Lötaugen 1, 2, 4 und 5 auf der Hauptplatine (Foto rechts) verbunden werden. Bei den Brücken 1 - 6 und 3 - 4 am Schalter isolierten Draht verwenden!

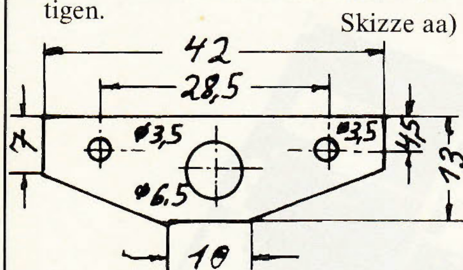
Skizze ee)
Pinbelegung



Die vier Schrauben des 3"-Laufwerkes lösen und beide Stecker abziehen.

3. Ein Stück Blech laut Skizze aa) anfertigen.

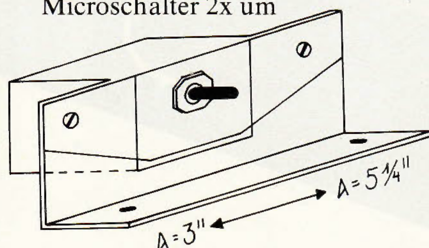
Skizze aa)



4. POWER-Schalter rechts im CPC einschließlich Haltewinkel abschrauben und Stecker abziehen. An dieser Stelle muß nunmehr eine Kurzschlußbrücke eingesteckt werden.
Der Haltewinkel wird nach Entfernen des Schalters nach unten nachgefeilt und zusammen mit dem Blech nach Skizze bb montiert (zwei Schrauben, 3mm x 10).

Skizze bb)

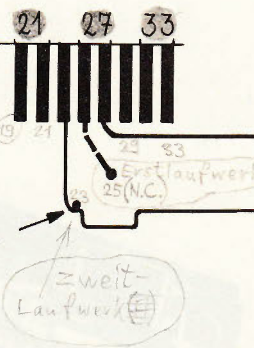
Microschalter 2x um



5. Platine des CPC von Unterseite abschrauben.

6. Platine umdrehen und fehlende Verbindung Laufwerk A zu Pin 25 des Platinensteckers (zweites Laufwerk) herstellen. Siehe hierzu Skizze cc.

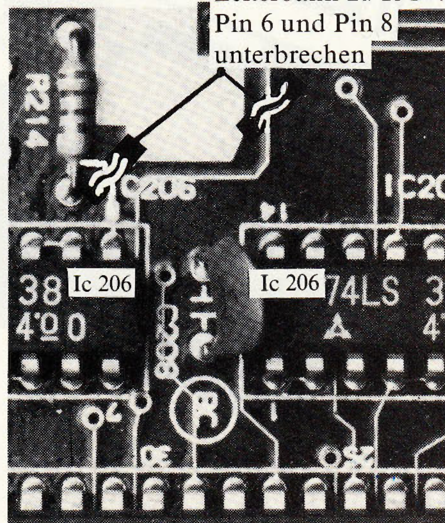
Skizze cc)



7. Platine wieder auf das CPC Unterteil schrauben.

8. Leiterbahn von IC 203 Pin 8 zu R214 (2.2 KOhm) unterbrechen. Leiterbahn von IC 203 Pin 6 zu Stecker 3"-Floppy unterbrechen. (Skizze dd)

Floppystecker oben



9. Nun werden die vier Drähte laut Skizze auf der Platine und am Schalter angelötet. Am Schalter wird jeweils über kreuz eine Verbindung hergestellt. Wird der Schalter laut Skizze ee verdrahtet und eingebaut, ist bei Schalterstellung nach vorne die 3"-Floppy im definierten Zustand A.

10. Alle Verbindungen sorgfältig prüfen, danach CPC zusammenbauen und verschrauben.

Ob die Spannungen zum 5 ¼"-Laufwerk vom CPC genommen werden oder mit eigenem Netzgerät versehen werden, sei jedem selbst überlassen. Letzteres wäre auf jeden Fall besser, um Kopplungen von vornherein zu unterbinden bzw. die Reserve für spätere Einbauten wie Speichererweiterung, serielle Schnittstelle etc aufzuheben. Als Bezugsquelle für eigene Gehäuseerstellung sei auf die Firma 'isert' in Eiterfeld verwiesen.

Um mit dem 5 ¼"-Laufwerk zu arbeiten, müssen wir uns noch eine 5 ¼"-Systemdiskette erstellen. Dazu wird der Schalter nach vorne gelegt ('definiertes Laufwerk 3" = A u. 5 ¼" = B). Anschließend Systemdiskette in das 3"-Laufwerk schieben und mit COPYDISC die Übertragung auf das Laufwerk B vornehmen.

Nach diesem Vorgang Schalter nach hinten legen, wodurch jetzt die 5 ¼"-Floppy unter Laufwerk A definiert ist. Es können jetzt Disketten formatiert und Duplikate erstellt werden. Es steht nunmehr nichts mehr im Wege, Software auf den 5 ¼"-Laufwerken zu lesen und zu beschreiben, sofern diese dem Format entsprechen. Hans Ziegler

Literaturverzeichnis

Titel	Inhalt/Zielgruppe	Verlag
Maschinensprache		
Z-80 Assembler-Handbuch	Nachschlagewerk für MC-Programmierer	Hofacker, DM 29,80
Z-80 Anwendungen	professionelle Anwender, Fortgeschrittene	Sybex-Verlag, DM 48,-
Programmieren in Maschinensprache mit Z-80	Anfänger und Fortgeschrittene	Hofacker, DM 39,-
Programmierung des Z-80	Anfänger, Fortgeschrittene, Profis	Sybex-Verlag, DM 58,-
CPC 464 Programmieren in Maschinensprache	Einsteiger	Markt + Technik, DM 46,-
Das Maschinensprachebuch zum CPC	Einsteiger	Data Becker, DM 39,-
Das Prozessorbuch zum Z-80	Technik + Programmierung	Data Becker, DM 59,-
Hardware:		
CPC Hardware-Erweiterungen	Bastler, Elektroniker	Data Becker, DM 49,-
Hardware-Handbook	Hardwarespezialisten	Hofacker, DM 49,-
CP/M		
CP/M-Trainingsbuch	CP/M 2.2 und 3.0	Data Becker, DM 49,-
CP/M 2.2 Anwenderhandbuch	Einsteiger und Fortgeschrittene	Markt + Technik, DM 46,-
Programmentwicklung unter CP/M 2.2	Programmierhilfen	Markt + Technik, DM 52,-
CP/M Plus	Einsteiger, Fortgeschrittene	IWT-Verlag, DM 48,-
CP/M User Guide 2.2	CP/M-Manual (engl.)	Schneider Data, DM 59,-
Logo:		
Grundkurs in Logo	Schüler, Studenten, Hobbyprogrammierer, Lehrer	Sybex-Verlag, DM 29,80
Logo - Computersprache für Eltern und Kinder	Anfänger	TE-WI-Verlag, DM 59,-
Logo - selbst gelernt	Anfänger	MVG-Verlag, DM 29,80
Start mit Logo auf CPC 464/664	Anfänger	Vogel Verlag, DM 30,-
Dr. Logo auf dem CPC	Einsteiger und Fortgeschrittene	Markt + Technik, DM 46,-
Guide to Logo	Einsteiger und Fortgeschrittene (engl.)	Schneider Data, DM 69,-
Floppy:		
Das große Floppy-Buch	Einsteiger, Fortgeschrittene, Profis	Data Becker, DM 49,-
DDI-1 Firmware Manual	ROM-Routinen der Schneider-Floppy	Schneider Data, DM 79,-
Allgemeine Bücher:		
Handbuch zum CPC	Ergänzung zum Handbuch	Du Mont, DM 24,80
Schneider CPC 464/664 Anwenderprogramme 1	Programmsammlung	Birkhäuser, DM 34,-
Arbeiten mit Schneider CPC	effektives Programmieren	Sybex-Verlag, DM 38,-
Mein Schneider CPC	Ergänzung zum Handbuch	Sybex-Verlag, DM 42,-
Schneider CPC dBase II	dBase II Anleitung + Programmsammlung	IWT-Verlag, DM 48,-
Schneider CPC Grafik	Einführung, Beispiele, Anwendungen	IWT-Verlag, DM 48,-
CPC 6128/664 Praxisbuch	Einsteiger, Ergänzung zum Handbuch	Signum Medien Verlag, DM 29,80
CPC 464/664 Praxis	Schwerpunkt Grafik	Hüthig-Verlag, DM 34,-
Schneider CPC Mathematik	Programme für den Anwender	IWT-Verlag, DM 44,-
CPC 464 Tips + Tricks	Tips + Programme	Data Becker, DM 39,-
CPC 464 intern	dok. ROM-Listing	Data Becker, DM 69,-
Standard-Basic Buch	Basic-Lehrgang	Heim-Verlag, DM 68,-
Basic - leicht und schnell gelernt	Einsteiger	Heim-Verlag, DM 68,-
CPC 464 für Einsteiger	Ergänzung zum Handbuch	Data Becker, DM 29,-
Basic-Lexikon zum CPC	Einsteiger	Heim-Verlag, DM 58,-

Titel	Inhalt/Zielgruppe	Verlag
Allgemeine Bücher		
Schneider 464 - Mein erstes Basic-Programm	Basic-Einsteiger	Sybex-Verlag, DM 32,-
Basic-Trainingsbuch zum 464	Ergänzung zum Handbuch	Data Becker, DM 39,-
CPC 464 Grafik + Sound	Schwerpunkt Grafik	Data Becker, DM 39,-
Schneider CPC Wirtschaft	Programme für den Anwender	IWT-Verlag, DM 48,-
CPC 464 inside out	ROM + Firmware-Listing	Huslik, DM 59,-
Peeks + Pokes zum CPC	Fortgeschrittene, Basic und Maschinensprache	Data Becker, DM 29,-
Basic-Brevier für CPC 464	Einführung in die Basicprogrammierung	Heise-Verlag, DM 29,80
Praktische Programme für den CPC	Programme für Heim und Hobby	MVG-Verlag, DM 29,80
Mathematik auf dem CPC	Problemlösungen in Basic	Heim-Verlag, DM 49,-
Höhere Mathematik auf dem CPC	Problemlösungen in Baisic	Vogel-Verlag, DM 33,-
Superspiele und Utilities für 464/664	Spiele- und Routinensammlung	Vogel-Verlag, DM 33,-
Start in die künstliche Intelligenz mit CPC 464	Vermittlung von Programmiertechniken	Vogel-Verlag, DM 23,-
CPC 6128 für Einsteiger	Ergänzung zum Handbuch	Data Becker, DM 29,-
CPC Tips + Tricks Band 2	Tips für alle CPC-Rechner	Data Becker, DM 39,-
CPC 664/6128 intern	ROM-Routinen	Data Becker, DM 69,-
Ideenbuch zum CPC	Programme für Heim und Hobby	Data Becker, DM 39,-
DFÜ für jedermann	Grundlagen, Techniken und Anwendungen	Data Becker, DM 39,-
ROM-Listing CPC 464/664/6128	dokum. ROM-Listing	Markt + Technik, DM 64,-
Wordstar für CPC	Ergänzung zum Handbuch	Markt + Technik, DM 49,-
Schneider CPC Systembuch	Nachschlagewerk für alle CPC-Anwender	Sybex-Verlag, DM 48,-
Schneider CPC Grafikbuch	Einführung in die Grafikprogrammierung	Sybex-Verlag, DM 42,-
Grafik auf dem CPC 464	Einführung in die Grafikprogrammierung	Westermann Verlag
CPC 464/664 Praxis II	Datenverwaltung	Hüthig-Verlag, DM 34,-

Sie wollen uns ein Programm zusenden? Folgende Regeln sollten Sie beachten:

1) Schicken Sie Ihre Programme auf Kassette oder Diskette. Dabei mehrmals hintereinander abspeichern.

2) Legen Sie eine ausführliche Programmbeschreibung bei, aus der eindeutig der Sinn und Zweck des Programmes hervorgeht. Eine Variablenliste ist sehr wichtig, in ihr sollten alle im Programm definierten Variablen mit ihrer Verwendung aufgeführt sein.
Beispiel: n$ = Nachname
g = glücklicher usw.

3) Wenn Sie gleichzeitig Besitzer eines Druckers sind, schicken Sie auf jeden Fall ein Programmlisting mit.

4) Vergessen Sie Ihren Absender nicht! Vollständige Anschrift, wenn möglich auch die Telefonnummer, ersparen unnötige Verzögerungen bei eventuellen Nachfragen unsererseits.

5) Behalten Sie auf jeden Fall eine Kopie Ihres Programmes, unvorhersehbare Umstände, wie z.B. Bandriß der Kassette, werden somit nicht zur Katastrophe.

6) Wenn Sie sogar mehrere Programme geschrieben haben und diese veröffentlichen wollen, nehmen Sie bitte für jedes Programm eine separate Kassette (mehrmals abspeichern!).

Damit erleichtern Sie unsere Arbeit wesentlich und eine schnellere Bearbeitung wird möglich.

7) Versehen Sie das Programm mit den sehr wichtigen REM-Statements. Das fördert nicht nur die Übersicht für uns und unsere Leser, sondern unterstützt zusätzlich das strukturierte Programmieren. Merke: Auch erfahrene Programmierer verwenden REM-Zeilen, das Programm sollte schließlich jeder verstehen und anwenden können.

8) Vermeiden Sie möglichst Variablennamen, wo leicht l und i oder 0 und o verwechselt werden können. Variablen sollten grundsätzlich in Kleinbuchstaben geschrieben werden und möglichst verständlich sein. Die Variable butter sagt mehr aus, als nur bu.

9) Bieten Sie das Programm niemals verschiedenen Verlagen gleichzeitig an! Mit der Veröffentlichung und dem daraus resultierenden Honorar, gehen die Urheberrechte an den Verlag über! Sollte Ihr Programm dann noch in einer anderen Zeitschrift abgedruckt werden, liegt ein Verstoß gegen das Wettbewerbsrecht vor.

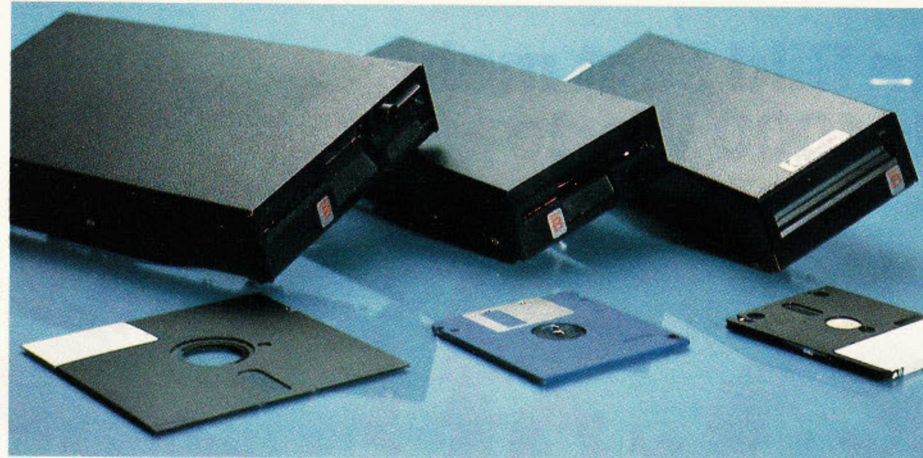
10) Jedes bei uns ordnungsgemäß eingegangene Programm wird ausführlich begutachtet und getestet. Sie erhalten von uns dann umgehend Bescheid, ob Ihr Programm veröffentlicht wird oder nicht. Wenn Sie das Programm nach unserer Begutachtung zurückhaben wollen, legen Sie bitte der Einsendung einen frankierten und adressierten Rückumschlag bei. Die Kosten einer eventuellen Rücksendung sind in jedem Fall vom Einsender zu tragen.

Falls wir uns für den Abdruck Ihres Programmes entschieden haben, setzen wir uns unverzüglich mit Ihnen in Verbindung und besprechen Ihr verdientes Honorar!
Übrigens wäre es nett, wenn Sie uns auch ein paar persönliche Daten schreiben würden (z.B. Ihr Alter, Tätigkeit, wie und wann kam die Idee zu dem Programm, was alles machen Sie mit dem CPC usw.?). So, jetzt aber genug der Theorie, mit fast unerträglicher Spannung und Neugier wartet Ihre „Schneider CPC international" auf Ihre Beiträge.

Einsendungen bitte an: DMV - Daten & Medien Verlagsges.mbH, Fuldaer Straße 6, 3440 Eschwege

3,5 Zoll am CPC

Bauanleitung für den Anschluß eines 3,5 Zoll-Zweitlaufwerkes für Schneider CPC 464/664/6128



1. Seit einiger Zeit sind preisgünstige Diskettenlaufwerke aus Restbeständen, Überproduktionen usw. zu erwerben. Ein Grund für Hobbybastler, sich Gedanken darüber zu machen, ob sie bei diesen Preisen nicht ein zweites Laufwerk an ihren Computer anschließen sollten. Im Falle der Schneider Computer gibt es hier an sich überhaupt keine Frage nach einem sinnvollen Einsatz dieses Zweitlaufwerkes, da speziell bei ernsthafter Anwendung dieses Computers, ein zweites Laufwerk unverzichtbar ist. Allein, wenn man an die dauernde Diskettenwechselei beim Kopieren von Programmen oder Disketten denkt, und allein das Programm »Multiplan« mit seinem umfangreichen Help-File, fast eine gesamte Diskettenseite belegt, ist leicht ersichtlich, daß man die Effektivität seiner Arbeit am Computer durch den Einsatz eines zweiten Laufwerkes sicherlich noch steigern könnte. In der Regel sind jedoch Computeranwender und Computerinteressierte nicht notwendigerweise mit Elektronikbasteleien vertraut und es ergibt sich eine durchaus natürliche Hemmschwelle, die sie daran hindert, sich ein solches Zweitlaufwerk anzuschließen. Für diese Leute ergibt sich durch die Möglichkeit, nahezu alle Komponenten die benötigt werden, um ein solches Zweitlaufwerk am Schneider CPC anzuschließen, fertig zu kaufen, dennoch die Möglichkeit zu einem lauffähigen Zweitlaufwerk für ihren Computer zu gelangen.

Obwohl dann nahezu alle Teile kommerziell gefertigt sind, ergibt sich, je nach Einkaufspreis des Laufwerkes und des entsprechenden Zubehörs, immer noch eine Ersparnis gegenüber einem kommerziell gefertigten Gerät. In diesem Falle wäre nur ein Stromversorgungskabel selbst herzustellen, und die gesamte Arbeit bei der Herstellung des Floppylaufwerkes würde sich auf mechanische Montage des Laufwerkes in einem vorgefertigten Gehäuse und das Herstellen von Verbindungen zum Computer mit Hilfe vorgefertigter Kabel, beschränken.

Für diejenigen, die mit Elektronikbasteleien etwas vertraut sind, sind die Möglichkeiten zu sparen sicherlich noch etwas größer, da sie bestimmte Teile, die für den Betrieb eines zweiten Laufwerkes erforderlich sind, in Eigenleistung erstellen werden.

2. Doch nun genug der Vorrede, lassen Sie uns zur Sache kommen. Wenn wir beabsichtigen, ein solches Zweitlaufwerk an einen vorhandenen Floppycontroller anzuschließen, müssen wir zunächst einmal wissen, welche Daten ein solches Zweitlaufwerk haben muß, um sich am Schneider-Floppycontroller betreiben zu lassen. Wenn man das Anschlußbild für das zweite Diskettenlaufwerk betrachtet, stellt man fest, daß alle Leitungen, die herausgeführt sind, für ein zweites Diskettenlaufwerk (36-polige Buchse bzw. Steckerleiste an der Rückseite des Gerätes), einer bekannten Norm für Diskettenlaufwerke entsprechen, dem sogenannten »Shugart-Bus« (Abbildung 1). Deshalb ist es grundsätzlich möglich, alle Diskettenlaufwerke, die über derartige »Anschlußleitungen«, eben den besagten Shugart-Bus verfügen, an einen Schneider Computer anzuschließen. Hierbei sollte man zusätzlich zu dem Vorhandensein des Shugart-Busses noch darauf achten, daß die Step-Rate-Time, des in Aussicht genommenen Laufwerkes, möglichst gering ist. Im DOS (Disk Operating System), dem Teil des Computers, der für alles, was mit Diskettenlaufwerken zu tun hat, zuständig ist, sind hier nach dem Einschalten Werte von ca. 12 Millisekunden für die Step-Rate-Time vorgegeben, d.h. die Zeit, die der Kopf des Diskettenlaufwerkes für den Transport von einer Spur zur nächsten benötigt. Liegt die Step-Rate-Time bei oder unter diesem Wert, ist ein problemloser Betrieb des Laufwerkes ziemlich sicher. Ist das jedoch nicht der Fall, sollte man die Flinte auch noch nicht »ins Korn werfen«. Die Einstellung dieser Zeit läßt sich auch softwaremäßig verändern. Das läßt sich einmal unter CP/M mit dem Programm "SETUP.COM" erreichen, unter AMSDOS muß mit einem kleinen Hilfsprogramm für eine richtige Einstellung der vorgegebenen Werte gesorgt werden. Darüber hinaus sollte man sich vergewissern, daß das in Augenschein genommene Floppylaufwerk auch ein "Ready"-Signal an den Computer zurückmeldet, da der Controller es abfragt und für die Steuerung verwenden wird. Ist das nicht der Fall, läßt sich dieses Problem jedoch auch mit Hilfe einer Brücke im Computer lösen, da die Amstrad-Entwickler derartige Schwierigkeiten wohl vorausgesehen haben. Mit Hilfe dieser Brücke wird das "Ready"-Signal im Computer künstlich erzeugt, und dem Betrieb auch eines solchen Laufwerkes steht dann nichts mehr im Wege.

Außerdem ist es jetzt an der Zeit, einige grundsätzliche Daten von Diskettenlaufwerken zu besprechen. Für einige Leser werden dies sicherlich »alte Kamellen« sein, für die anderen sind sie jedoch sehr hilfreich bei der Auswahl des in Frage kommenden Laufwerkes. Man unterscheidet bei Diskettenlaufwerken zwischen solchen mit einfacher und doppelter Schreibdichte (Single- oder Double-Density). Auf einen einfachen Nenner gebracht bedeutet das, daß ein Laufwerk mit doppelter Schreibdichte doppelt so viel Daten speichern kann wie eins mit einfacher Schreibdichte, da die Fläche, die zum Beschreiben zur Verfügung steht, statt in 40 Spuren z.B. in 80 Spuren aufgeteilt wird. Damit ergibt sich eine doppelt so hohe Kapazität der Diskette, ähnlich wie bei einem Tonbandgerät, das anstelle von zwei Tonspuren mit vier arbeitet und damit die doppelte Spielzeit erzielt. Für unsere Zwecke sind Diskettenlaufwerke mit einfacher Schreibdichte ausreichend, da das DOS der Schneider Computer nicht mehr als 40 Spuren zuläßt. Aus dem gleichen Grund ist ein einseitiges Laufwerk ausreichend, da auch nur einseitige Laufwerke unterstützt werden.

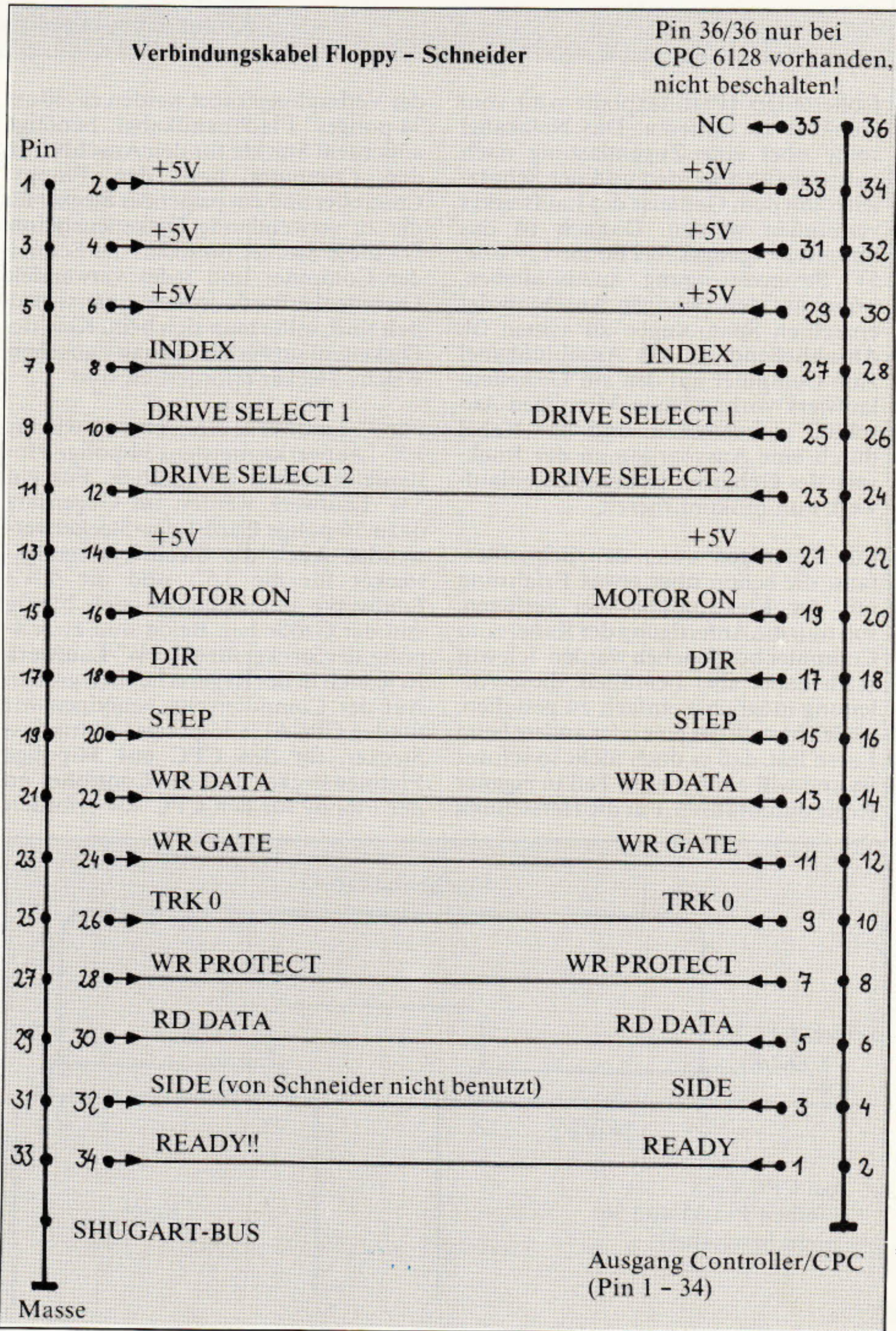
Nehmen wir einmal an, ein preisgünstiges Laufwerk wäre in Aussicht, es hätte die geforderten Daten (nahezu jedes moderne Laufwerk weist sie auf, lediglich bei »Oldtimern« ist Vorsicht geboten). Was wird dann außer dem eigentlichen Laufwerk noch benötigt? Hierzu möchte ich Ihnen anhand einer kleinen Liste Auskunft geben.

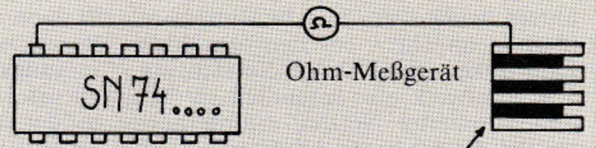
1 Diskettenlaufwerk (Shugart-Bus)
1 Netzteil (5/12 Volt)
1 Stromversorgungsstecker für Floppy
1 Anschlußkabel (Floppy-Computer)
1 10 cm vierpoliges Kabel
1 Floppygehäuse

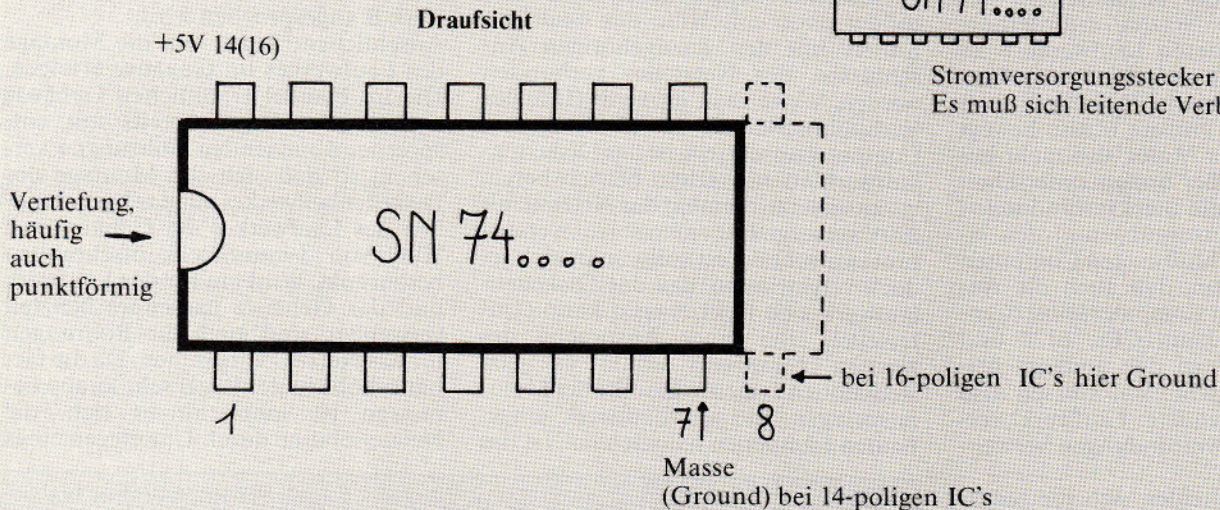
Bewußt ist bis jetzt noch nicht die Rede von 3"-, 5¼"- oder 3,5"-Laufwerken. Diese Bauanleitung hat nämlich grundsätzlich Gültigkeit für alle Laufwerkstypen, wurde jedoch mit einem »Epson 3,5"-Diskettenlaufwerk (SMD-110)« erprobt. Diskettenlaufwerke – gleich welchen Formats – unterscheiden sich nämlich, soweit es das Lesen und Schreiben von Daten angeht, nur unwesentlich – wenigstens was den Teil angeht, der uns interessiert. Natürlich gehe ich davon aus, daß Sie über den

Shugart-Bus verfügen. Eines sollte man jedoch zusätzlich noch wissen. Das Netzteil, das verwendet werden muß, unterscheidet sich – was die Dimensionierung angeht – bei den einzelnen Laufwerken grundsätzlich. So ist für ein 5 ¼"-Laufwerk ein wesentlich stärker dimensioniertes Netzteil erforderlich, als für die kleineren Laufwerke – an sich verständlich, wenn man einmal bedenkt, daß hier mit der Diskette ja auch eine wesentlich größere Masse bewegt werden muß, der Motor also mehr leisten muß. Darüber hinaus unterscheiden sich auch die Steckverbindungen für die Stromversorgung und den eigentlichen Anschluß an den Computer. Abgesehen davon, daß auch die Anschlußstecker am Computer selbst (Centronics-Stecker beim CPC 6128 und Platinenstecker bei den anderen beiden) sich unterscheiden, müssen also für jedes verwendete Laufwerk verschiedene Steckverbindungen verwendet werden.
Hierbei unterscheiden sich die Steckverbindungen für das Schneider 3"- und einige 3,5"-Laufwerke nicht, so daß beim Anschluß einer solchen 3,5"-Floppy auf ein Original Schneider Anschlußkabel zurückgegriffen werden kann und sich speziell für ungeübte Computerfans die Anfertigung eines passenden Anschlußkabels Computer-Diskettenlaufwerk bzw. Controller-Diskettenlaufwerk, erübrigt, bzw. zum Ausprobieren zunächst einmal verwendet werden kann. Doch zur Anfertigung dieser Kabel soll später noch einmal Stellung genommen werden. Gehen wir davon aus: Der passende Stromversorgungsstecker sei besorgt, das Kabel zum Anschluß an den Computer liege vor, und auch das Netzteil und das passende Gehäuse sei vorhanden.

3. Als nächstes gilt es herauszufinden, welche Anschlüsse der Stromversorgung mit 12 Volt, welche mit Masse und welche mit 5 Volt zu verbinden sind. Dazu benötigt man ein einfaches Vielfachmeßinstrument mit einem Ohm-Meßbereich. Der oder die Masseanschlüsse des Diskettenlaufwerkes lassen sich leicht herausfinden, indem man eine Spitze des Meßinstrumentes an das Chassis des Laufwerkes hält und mit der anderen die vier Anschlußpunkte der Stromversorgung abtastet. Dort, wo sich kein Widerstand zeigt, ist eine Masseverbindung der Stromversorgung. An diese Punkte wird später die Masse des Netzteiles gelegt. Auch der 5 Volt-Anschluß ist leicht zu finden. Auf der Platine mit der Steuereinheit für das Diskettenlaufwerk sind mehrere integrierte Schaltkreise mit Bezeichnungen, die mit den Buchstaben 'SN' beginnen (siehe Skizze) zu sehen. Am Anschluß 7 dieser Schaltkreise bei 14-poligen Schaltkreisen bzw. Anschluß 8 bei 16-poligen, ist immer Masse zu finden, während am Anschluß 14 bzw. 16 die Versorgungsspannung von 5 Volt anliegt. Ist z.B. zwischen Anschluß 14 und einem Anschluß für die Stromversorgung kein Widerstand festzustellen, kann man davon ausgehen, den 5-Volt-Anschluß des Diskettenlaufwerkes gefunden zu haben. Der 12-Volt-Anschluß ist dann der übrige Anschluß, da in der Regel zwei Anschlüsse des Stromversorgungssteckers Masse führen.
Diese solcher Art gefundenen Anschlüsse sollten dann mit Hilfe des 4-adrigen Kabels mit den dazugehörigen Anschlüssen des Netzteiles verbunden werden. Wenn man jetzt mit Hilfe des Verbindungskabels das Laufwerk am Computer anschließt, ist das Diskettenlaufwerk an sich schon betriebsbereit, vorausgesetzt, man hat das Netzteil mit 220V versorgt. Für einen solchen ersten provisorischen Test sollte man sich jedoch vergewissern, daß das Diskettenlaufwerk sich frei bewegen kann (Unterseite!). Bei einer Ansprache des zweiten Laufwerks sollte es jetzt Reaktionen zeigen, z.B. sollte der Motor anspringen, die am Laufwerk eingebauten Lämpchen leuchten u.ä. Ist das nicht der Fall, ist wahrscheinlich eine Brücke auf der Steuerplatine des Diskettenlaufwerkes nicht richtig gesetzt. Mit dieser Brücke wird eingestellt, ob das Laufwerk als Laufwerk B, C usw. arbeitet. Die Brücke sollte so gesteckt sein, daß unser Laufwerk als Laufwerk B angesprochen wird.
Anschließend kann nun die Montage des Laufwerkes im Gehäuse erfolgen. Die im Handel erhältlichen Gehäuse sind in der Regel bereits mit entsprechend passenden Bohrungen versehen, so daß sich die Montage des Diskettenlaufwerkes auf das Festschrauben des Laufwerkes am Gehäuse mit Hilfe von 3-mm-Steckkopfschrauben beschränkt. Wird ein für das Laufwerk und das Gehäuse passendes Netzteil verwendet, sind auch die Bohrungen für das Netzteil vorgesehen. Da die Gehäusehöhe in der Regel sehr knapp bemessen ist, empfiehlt es sich, die Netzteilplatine durch Unterlegen eines



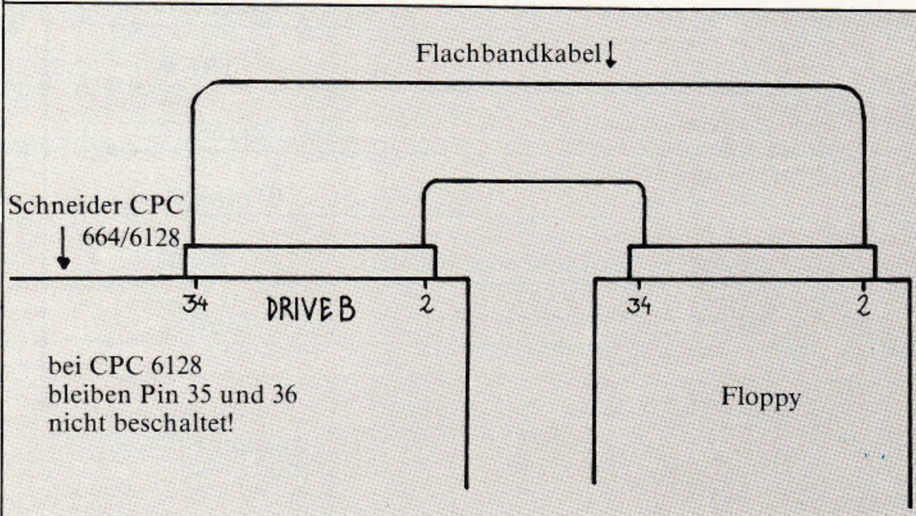




Isolierteilers (Pertinaxplatte o.ä.) vom Gehäuse zu isolieren. Das Netzkabel sollte über eine Zugentlastung nach außen geführt werden und der Schutzleiter mit dem Gehäuse des Laufwerkes verbunden werden. Danach ist das Stromversorgungskabel für die 12V bzw. 5V Stromversorgung anzuschließen, wobei auf den richtigen Anschluß der einzelnen Spannungen zu achten ist. Anschließend ist das Anschlußkabel zum Computer auf den am Diskettenlaufwerk vorhandenen 34-poligen Anschluß aufzustecken, und das Kabel durch eine Aussparung an der Rückseite des Gehäuses ist unser Zweitlaufwerk nun funktionsbereit.

Für diejenigen unter den Schneiderfans, die schon über etwas Erfahrung mit der Elektronikbastelei verfügen, soll nun die Anfertigung der Kabel zum Computer beschrieben werden. Ich will mich dabei aber bemühen, diese Anleitung möglichst einfach zu gestalten, vielleicht stellt der eine oder andere beim Lesen fest, daß es doch nicht so schwer ist und will auch diesen Teil in eigener Regie durchführen. Für die Herstellung der Verbindungskabel werden ca. 30 cm 34-poliges Flachbandkabel benötigt, außerdem Stecker für den Anschluß an den Computer bzw. den Floppy-Controller und ein passender Stecker für die zu verwendende Diskettenstation. Da diese Stecker für jeden der Schneider Computer und jedes verwendete Diskettenlaufwerk jedoch unterschiedlich sind, sollte man sich beim Kauf des Diskettenlaufwerkes genau ansehen, welche Stecker erforderlich sind.

Für 3"-Laufwerke sind, ebenso wie für 3,5"-Laufwerke, meistens 34-polige Pfostenfeldbuchsen erforderlich. Für ein 5 ¼"-Laufwerk werden im Gegensatz dazu 34-polige Card-Edge-Stecker verwendet. Auch die Stromversorgungsstecker für die 3,5"- und die 5 ¼"-Laufwerke unterscheiden sich häufig von der Größe her. Beide sind zwar 4-polig, der Stecker für das 5 ¼"-Laufwerk ist jedoch in der Regel erheblich größer. Auf der Computerseite benötigen wir für den CPC 6128 36-polige Centronics-Stecker, für den CPC 664 34-polige Platinenstecker und der Controller an der Floppy für den CPC 464 erfordert wiederum 34-polige Platinenstecker. Alle diese Stecker sollte man in einer Ausführung zum Klemmen besorgen, da wir uns bei der hohen Übertragungsrate des Laufwerkes schon im Bereich von Hochfrequenzen bewegen, und schlechte Übergänge an Lötstellen eine Fehlfunktion verursachen könnten. Außerdem sollte auch das Flachbandkabel mechanisch nicht über die Gebühr beansprucht werden, z.B. durch Knickungen usw. Um ein zweites Laufwerk richtig am Schneider Computer anschließen zu können, sollte man ebenfalls wissen, daß alle Leitungen des Shugart-Busses am Schneider Computer für den Anschluß eines zweiten Laufwerkes spiegelbildlich ausgegeben werden (siehe Anschlußbild und vergl. Computerhandbuch). Dies sollte es bei der Herstellung des Kabels beachtet werden. Zunächst einmal sind also die Anschlüsse zwischen Computer und Diskettenlaufwerk spiegelgleich zu gestalten, d.h., die Seite des Kabels, die am Computer rechts herauskommt, führt zur linken Seite des Diskettenlaufwerkes, und umgekehrt (siehe Skizze). Nachdem dann das Flachbandkabel richtig in die Vertiefungen zwischen den beiden Hälften des Klemmsteckers eingelegt ist, die einer Führung des Kabels dienen, drückt man diese zunächst einmal mit der Hand zusammen, um sie anschließend in einem Schraubstock o.ä. voll zusammenzupressen. Dabei sollte das Flachbandkabel möglichst wenig überstehen, da ansonsten die Gefahr besteht, daß mit dem Deckel des Gehäuses Kurzschlüsse auf dem Anschlußkabel zum Computer verursacht werden. Ähnlich verfährt man mit dem Stecker zum Anschluß an den Computer. Nachdem noch einmal überprüft wurde, ob alle Anschlüsse richtig getätigt worden sind, ist das Kabel schon fertig. In ähnlicher Weise lassen sich übrigens auch Druckeranschlußkabel, Kabel für ex-



terne Erweiterungen usw. anfertigen. Die Kabel für die Stromversorgung müssen in der Regel gelötet werden. Dabei empfiehlt sich die Verwendung eines Feinlötkolbens von max. 30 Watt, da sonst die Plastikführungen für die Anschlußstifte zu heiß werden und zu schmelzen beginnen. Ebenso sollte man die Lötzeit so kurz wie möglich halten, wenn man nicht eine Ausführung der Stromversorgungsstecker zur Verfügung hat, die eine spätere Montage der Anschlußstifte ermöglicht. In kommerziell gefertigten Netzteilen wird der Anschluß der Stromversorgungskabel meistens über Klemmsteine erfolgen, die auf der Platine befestigt sind und in die die Kabel dann einfach, wie in eine Lüsterklemme, eingeklemmt werden. Hierbei sind die Punkte mit Masse, 5-Volt und 12-Volt ganz einfach mit Hilfe eines Vielfachmeßinstrumentes im Spannungsbereich zu ermitteln oder dem beiliegenden Anschlußplan zu entnehmen. Leider kann hier keine allgemeingültige Bauanleitung für alle auf dem Markt befindlichen Diskettenlaufwerke gegeben werden, da jeder Hersteller sein »eigenes Süppchen kocht«, wie der Autor beim Vergleich von Diskettenlaufwerken verschiedener Hersteller leider feststellen mußte, so daß es durchaus zu Abweichungen von der geschilderten Bauart kommen kann, besonders was die Art der für Stromversorgung und Shugart-Bus verwendeten Stecker angeht. Die meisten dieser Stecker sind jedoch handelsüblich und von größeren Elektronikversendern zu beziehen.

Gleiches gilt auch für die Bauteile, die zur Erstellung des Netzteiles benötigt werden. Hierbei müssen nun zwei Versionen unterschieden werden, da die 5 ¼"-Floppy ja ein stärker dimensioniertes Netzteil benötigt. Für diejenigen, die ein solches Netzteil im Eigenbau erstellen wollen, hier ein Schaltbild mit Angaben zur Dimensionierung für die einzelnen Anwendungen und einer kurzen Funktionsbeschreibung (siehe Schaltbild).

Über eine Sicherung wird dem Transformator Strom zugeführt. Am Ausgang des Transformators stehen 2 x 9V Wechselspannung zur Verfügung, die jeweils in einem Brückengleichrichter in Gleichstrom umgewandelt werden. Die folgenden Kondensatoren glätten die Gleichspannung und anschließend gelangt diese dann auf zwei Spannungsregler, die für eine hochkonstante Ausgangsspannung von 12 bzw. 5 Volt sorgen. Die ein- und ausgangsseitig am Spannungsregler angebrachten Kondensatoren von 100 nF verhindern wilde Schwingungen des Spannungsreglers. Ein separates Netzteil für die Stromversorgung des Diskettenlaufwerkes ist unumgänglich, da die im Monitor eingebaute Stromversorgung nicht für den Anschluß eines zweiten Laufwerkes ausgelegt ist und »in die Knie gehen« würde. Beim Aufbau sollte man auf ausreichende Kühlung der Spannungsregler achten, da diese über eine integrierte Schutzschaltung verfügen, die bei zu hohen Temperaturen des Reglers diesen abschalten und das Diskettenlaufwerk dann nicht mehr arbeitet. Am besten schraubt man die Regler an das Gehäuse, in dem das Diskettenlaufwerk untergebracht wird, und verwendet dieses auf diese Art und Weise zur Ableitung der beim Betrieb entstehenden Wärme. Bei einigen Fertiggehäusen sind für diese Art der Wärmeableitung bereits Bohrungen vorgesehen.

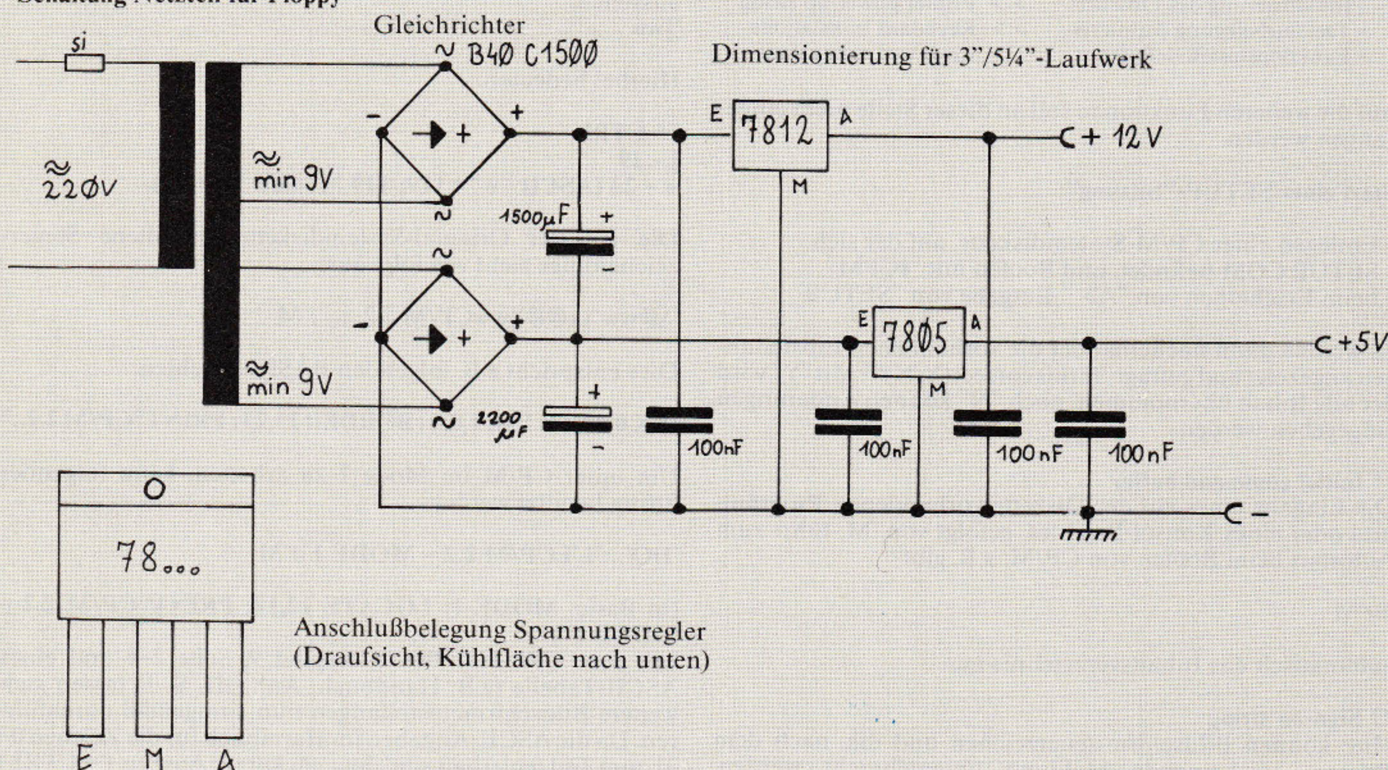
Benötigte Teile für das Netzteil:

1 Transformator prim.220V sec. 2 x9 V, je 0,5 – 0,8A (je 2A)
2 Gleichrichter B 40 C 1500 (B 40 C 3200)
1 Elektrolytkondensator 2200 Mikro-Farad/16V (4700 M.-Farad/16V)
1 Elektrolytkondensator 1500 Mikro-Farad/16V (2200 M.-Farad/16V)
4 Kondensatoren 100 nF
1 Spannungsregler UA 7805 (UA 78 S 05) o.ä.
1 Spannungsregler UA 7812 (UA 78 S 12) o.ä.

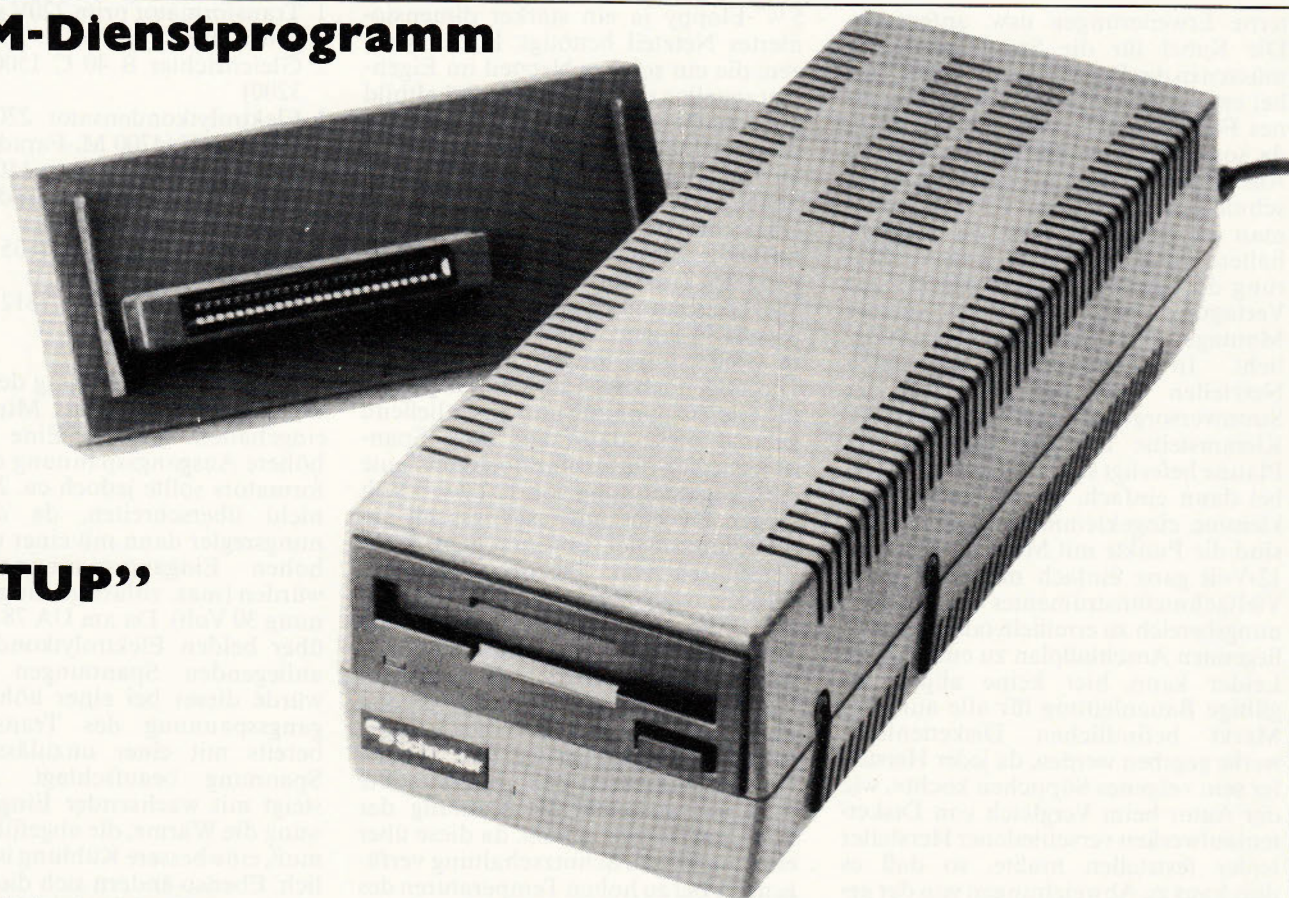
Für die Dimensionierung der Bauteile sollten diese Werte als Mindestwerte eingehalten werden. Eine eventuell höhere Ausgangsspannung des Transformators sollte jedoch ca. 2 x 10 Volt nicht überschreiten, da die Spannungsregler dann mit einer unzulässig hohen Eingangsspannung belastet würden (max. zulässige Eingangsspannung 30 Volt). Da am UA 7812 sich die über beiden Elektrolytkondensatoren anliegenden Spannungen addieren, würde dieser bei einer höheren Ausgangsspannung des Transformators bereits mit einer unzulässig hohen Spannung beaufschlagt. Außerdem steigt mit wachsender Eingangsspannung die Wärme, die abgeführt werden muß, eine bessere Kühlung ist erforderlich. Ebenso ändern sich die Werte für die Spannungsfestigkeit der Kondensatoren, da die an ihnen eingehende Spannung ca. 50 % höher als die Ausgangsspannung des Transformators ist. Bei einer Ausgangsspannung von 2 x 10 Volt erhält man so ca. 15 Volt an den Kondensatoren, diese sollten deshalb wenigstens bis zu dieser Spannung spannungsfest sein. Ein Aufbau des Netzteiles kann z.B. auf einer Lochrasterplatine erfolgen.

W. Wantia

Schaltung Netzteil für Floppy



CP/M-Dienstprogramm



"SETUP"

für 464-664



Im Handbuch für das Schneider Diskettenlaufwerk DDI-1, kommt die Beschreibung des nützlichen CP/M-Dienstprogrammes SETUP, wie vieles, leider zu kurz. Mit den folgenden Ergänzungen wird ein sinnvoller Gebrauch der Funktionen möglich, die beim ersten Laden (Booten) von CP/M aufgerufen werden:

- Autostart von Programmen >> Initial command buffer
- Initialisierung des Bildschirms >> Sign-on string
- Initialisierung des Druckers >> Printer power-up string
- Tastaturbelegung definieren >> Keyboard translations
- Funktionstasten definieren >> Keyboard expansions

Auf die weiteren Funktionen soll an dieser Stelle nicht eingegangen werden.

Start einer SETUP-"Sitzung":

- Einlegen einer CP/M-Systemdiskette, auf der sich SETUP.COM befindet, und Booten mit "|CPM"
- beim Erscheinen von "A> " Eingabe von "SETUP"

Es werden dann nacheinander die vorhandenen Einstellungen angezeigt und gefragt "Is this correct (Y/N):". Mit "Y" wird der alte Inhalt übernommen, nach "N" können Änderungen eingegeben werden.

**** Initial command buffer**
Die Eingabe eines auf der Diskette vorhandenen Programmes oder eines Konsol-Befehles, gefolgt von ^M, führt zum Autostart beim Booten von CP/M; z.B. gibt

dir^M

automatisch das Inhaltsverzeichnis aus.

**** Sign-on string**
Hier können Bildschirmsteuerzeichen und die, nach dem Booten, erscheinende Meldung untergebracht werden. Kontrollzeichenkommandos sind wie folgt zu verwenden:

Soll z.B. der Bildschirmrand (Border) in Farbe 23 erstrahlen, so muß dem Bildschirm folgende Steuerzeichenfolge übermittelt werden:

^29;23;23 (Kapitel 9/Seite 4 des 464-Handbuches)

Die Werte hinter dem Border-Steuerzeichen 29 werden dabei nach Modulo 32 ausgewertet, das heißt, es gilt der Restwert nach Teilung durch 32.

Im Sign-on-String kann die Steuerzeichenfolge nun wie folgt aussehen.
^]ww

Hierbei bedeuten:

^ - CTRL
] - 29
w - 23 (ASCII "w"=119; 119 Modulo 32 = 23)

Die auf der Original-Systemdiskette enthaltene Steuerzeichenfolge sieht wie folgt aus:

^\@ww^\a@@^]wwCP/M 2.2.......^J^M

Dies entspricht z.B. folgenden BASIC-Befehlen:

INK 0,23,23 : INK 1,0,0 : BORDER 23,23 : PRINT"CP/M 2.2.."

Um unter CP/M im Mode 1 zu arbeiten, kann folgender String benutzt werden:

^DQ^...^L^LCP/M 2.2 - MODE 1^J^M

(in Basic: **MODE 1: LOCATE 12,12: PRINT"CP/M 2.2 - MODE 1"**)
Mit dem CPC-Handbuch, Kapitel 9, Seite 2–4 und einer ASCII-Tabelle (z.B. Handbuch, Anh. III, S. 1) lassen sich weitere Bildschirmeinstellungen nun sinngemäß durchführen. Da die ASCII-Angaben im Handbuch für die Zeichen 0–31 zum Teil fehlerhaft sind, hier ein kurzer Auszug: (^=CTRL)

DEZ	ASCII	DEZ	ASCII	DEZ	ASCII	DEZ	ASCII
00	^@	08	^H	16	^P	24	^X
01	^A	09	^I	17	^Q	25	^Y
02	^B	10	^J	18	^R	26	^Z
03	^C	11	^K	19	^S	27	^[
04	^D	12	^L	20	^T	28	^\
05	^E	13	^M	21	^U	29	^]
06	^F	14	^N	22	^V	30	^>
07	^G	15	^O	23	^W	31	^....

**** Printer power-up string**
Hier können, entsprechend den Bildschirmsteuerzeichen, die gewünschten Druckersteuerzeichen untergebracht werden. Zur Wahl der NLQ-Schriftart beim NLQ 401 lautet z.B. der String:

^[x^A entspricht< ESC> "x"CHR$(1)

Für den verbreiteten Drucker Star SG 10 muß es so aussehen:

^[B^D entspricht< ESC> "B"CHR$(4)

Weitere Steuerzeichen sind sinngemäß möglich.

**** Keyboard translations**
Hiermit können den Tasten neue ASCII-Werte zugewiesen werden. Es wird für die Änderung folgendes Menue ausgegeben:
A - Eingabe neuer Tastenbelegung
D - Löschen der Tastenbelegung Nr. ( )
C - Löschen aller Tastenbelegungen
F - Eingabe beenden
Die Eingabe kann z.B. lauten:

c < ENTER> a,66,27,21,10 < ENTER> f< ENTER>

und belegen die ESC-Taste (66) wie folgt:

Normal	- 27	- ^[(ESC)
Shift	- 21	- ^U
Control	- 10	- ^J

Weiterhin sind in Verbindung mit der Eingabe von Expansionstrings max. 32 Expansion-Tokens mit den Nummern 128 - 159 möglich. Die kleine ENTER-Taste im Ziffernblock (Taste Nr. 6) kann z.B. wie folgt belegt werden:

a,6,128,128,128, < ENTER> f< ENTER>

und ruft dann bei Tastendruck den in den folgend beschriebenen Keyboard expansions abgelegten String auf.

**** Keyboard expansions**
Hier können Erweiterungsstrings abgelegt und auf Tastendruck aufgerufen werden. Soll z.B. ein Programm "HELP.COM" mit der oben belegten ENTER-Taste gestartet werden, so muß die Belegung lauten:

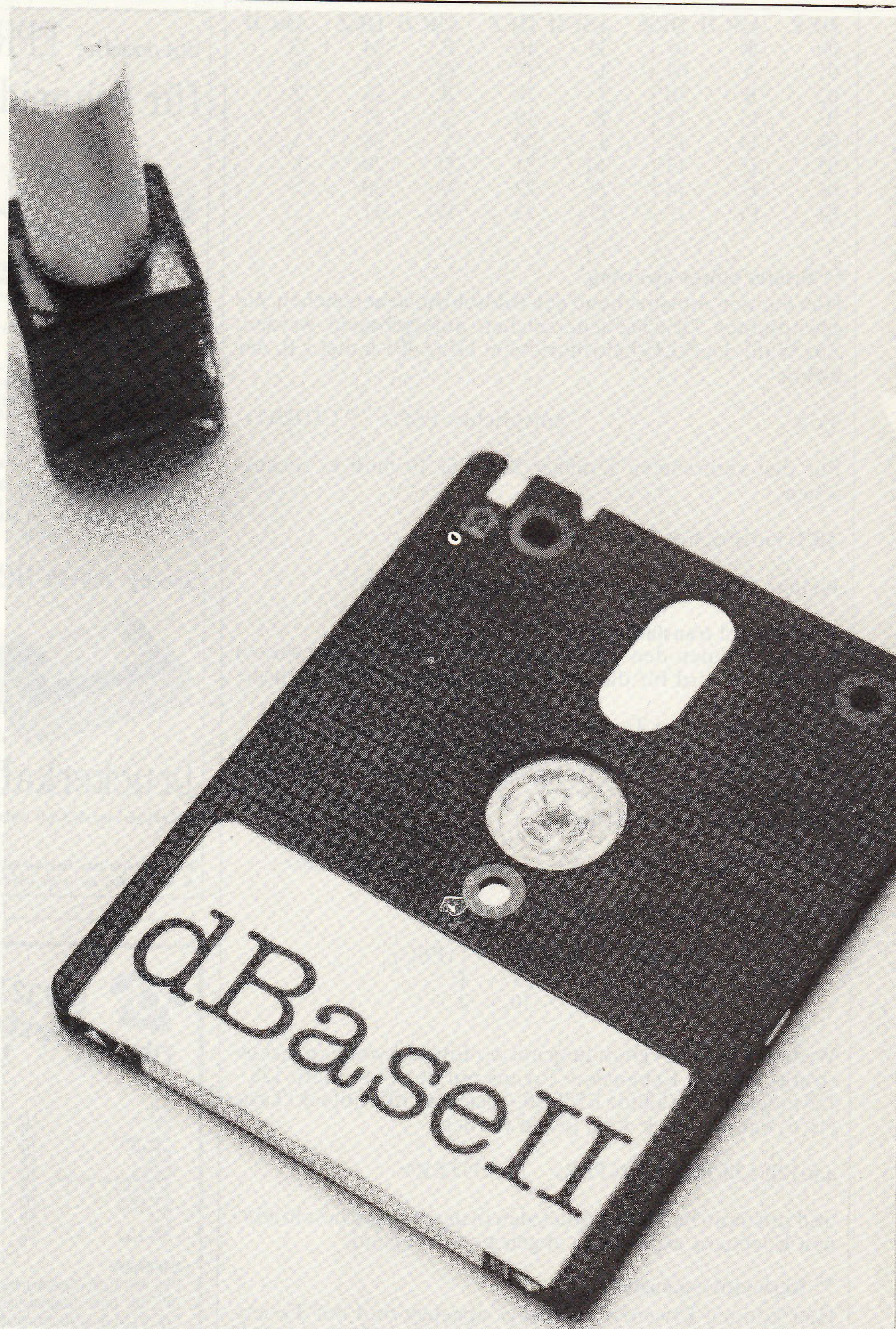
a,0,help^M < ENTER> f < ENTER>

Hierbei ist darauf zu achten, daß den bei den Keyboardtranslations benutzten ASCII-Nummern 128 - 159 hier die Token-Nummern 0-31 entsprechen.
Die folgenden Optionen sollen hier, wie gesagt, nicht weiter beschrieben werden (sie betreffen andere Schnittstellenfunktionen). Die entsprechenden Fragen sind mit "Y" zu beantworten, bis die Übernahme des neuen Konfigurations-Sektors erfolgt. Anschließend kann mit dem letzten "Y" ein CP/M-Kaltstart erfolgen und damit die geleistete Arbeit überprüft werden.
Mit den beschriebenen SETUP-Optionen lassen sich nahezu beliebige Systemeigenschaften erreichen, die automatisch nach dem Booten zur Verfügung stehen und das Arbeiten mit CP/M-Programmen wie z.B. auch WordStar oder dBase sehr komfortabel gestalten.

K.-P. Trenkel

Der letzte Schliff

dBASE II – Startdiskette für Schneider JOYCE



Endlich ist sie erhältlich, die JOYCE-Version von dBase II. Drucker- und Bildschirmanpassung sind gelöst, aber eine Kleinigkeit ist noch zu verbessern:

Man kann zwar mit der Sicherheitskopie schon arbeiten, hat aber lediglich maximal 49 KByte für Daten zur Verfügung, und dies ist oft zu wenig. Zwar kann man auf die HELP-Funktion verzichten, das File DBASEMSG.TXT löschen und erhält so nochmal 60 KByte Platz. Die beiden Systemfiles DBASE.COM und DBASEOVR.COM sind jedoch notwendig und belegen immer noch über 60K. Einfach dBase laden und dann Diskette wechseln geht also nicht. Oder doch?

Es geht!!!
dBase erwartet diese Files immer auf dem Laufwerk, von dem sie geladen wurden. Also braucht man sie nur in die glücklicherweise vorhandene RAM-Disk laden und von dort aus starten.

Für CP/M und dBASE wird die RAM-Disk mit "M:" zum Bezugslaufwerk erklärt, nach dem Start von dBASE wird mit "SET DEFAULT TO A" die normale Floppy für den Datenverkehr definiert. Zusammen mit wohldosierten RESETs bei Diskettenwechsel stehen nun volle 173 KByte für Daten zur Verfügung. Was will man mehr...

Tabelle 1

```
Directory For Drive A:  User  0
    Name      Bytes   Recs   Attributes         Name      Bytes   Recs   Attributes
 ------------ ------ ------ ------------    ------------ ------ ------ ------------
 ANLEIT   TXT    1k       2 Dir RW           DB       KEY    2k      16 Dir RW
 DB       SUB    1k       1 Dir RW           DBASE    COM   20k     158 Dir RW
 DBASEOVR COM   41k     328 Dir RW           DIR      COM   15k     114 Dir RW
 J12DCPM3 EMS   40k     320 Dir RW           PIP      COM    9k      68 Dir RW
 PROFILE  SUB    1k       1 Dir RW           SETKEYS  COM    2k      16 Dir RW
 SUBMIT   COM    6k      42 Dir RW           TYPE     COM    3k      24 Dir RW

 Total Bytes     =    141k  Total Records =    1090  Files Found =   12
 Total 1k Blocks =    141   Used/Max Dir Entries For Drive A:   17/  64
```

Man will mehr!!!
Jetzt sollte das Ganze noch möglichst schnell und von selbst geschehen, und außerdem mit weniger Diskettenwechsel. CP/M-Disk rein und booten, Diskette wechseln, dBASE laden, auf die Datendiskette umsteigen und endlich mit der Arbeit beginnen – das nervt.
Also muß eine Automatik her... und so geht's:

Bei jedem Start von CP/M wird eine eventuell vorhandene Befehlsdatei namens PROFILE.SUB ausgeführt, die das für uns erledigen kann.

Da wir auf die Sicherheitskopie (!!!) von dBASE auf alle Fälle das File SETKEYS.COM zur Tastaturdefinierung kopieren müssen, können wir gleich noch etwas mehr mitnehmen:

Löschen Sie auf der dBASE-Kopie (!) das Help-File mit "ERA DBASEMSG. TXT". Legen Sie die CP/M-Diskette ein und kopieren folgendes mit "PIP M:= (...)":

"PIP.COM"
"TYPE.COM"
"SETKEYS.COM"
"SUBMIT.COM"
"DIR.COM"
"J12DCPM3.COM"

Laden Sie Basic und erstellen Sie mit dem Editor RPED die beiden unten näher beschriebenen Files PROFILE.SUB und ANLEIT.TXT auf Laufwerk "m:' (Laufwerk im RPED-Menu ändern, auch die Filenamen müssen stimmen!!!). Verlassen Sie RPED mit 'EXIT'.

Geben Sie anschließend "M:" ein, drücken 'STOP' und wechseln auf die dBASE-Kopie (!).
Mit "PIP A:=*.*" werden die Daten jetzt auf die dBASE-Disk gespeichert und Ihre Startdiskette ist fertig!

Von jetzt an brauchen Sie nur noch den JOYCE einschalten, diese Diskette einlegen und dBASE ist sofort verfügbar. Alles was Sie noch beachten müssen, ist der RESET zur rechten Zeit und das SET DEFAULT TO A zu Anfang.

Schauen Sie sich jetzt Ihre dBASE-Diskette mit "DIRÄALLÜ" an; sie sollte wie in Tabelle 1 beschrieben aussehen:

Wenn dem nicht so ist, dann haben Sie irgendwo einen Fehler gemacht, also nochmal von vorne (oder entnervt auf Karteikästen umsteigen...).

Und hier sind die Files, die Sie selbst erstellen müssen:
(Bitte geben Sie sie genau ein!!!)

Das erste erhält den Namen **PROFILE.SUB:**

```
setkeys db.key
pip
<m:=dbase.com
<m:=dbaseovr.com
<
type anleit.txt
m:
dbase
```

Hiermit werden automatisch beim Start die Tastatur umdefininiert, die dBASE-Files kopiert und richtig gestartet.

Gleichzeitig wird mit "TYPE ANLEIT.TXT" ein kurzer Kommentar gedruckt, der Sie an die weiteren Schritte erinnern soll. Diesen Teil können Sie nach Belieben gestalten oder auch ganz weglassen (dann aber auch "TYPE ANLEIT.TXT" im PROFILE-Programm weglassen!!).

Mein Vorschlag für **ANLEIT.TXT:**

```
Startanleitung wie folgt eingeben:

. (Datum)
. reset
. set default to a

dann Datendiskette
einlegen und weiter
. reset ... etc.
```

Viel Vergnügen mit Ihrem "neuen' dBASE II.

Michael Anton

WORDSTAR auf dem Schneider CPC

Der Schneider CPC trägt in seinem Namen den stolzen Titel »Personal Computer«. Mit 64K Speicherplatz, 80 Zeichen auf dem Bildschirm und dem Diskettenbetriebssystem CP/M ist dies wohl auch nicht übertrieben, wenn da nicht das 3"-Diskettenlaufwerk wäre. Die relativ langsame Geschwindigkeit, die bloß 180K Diskettenkapazität, die einseitige Diskettenbenutzung und die Abweichung von jeglichem Diskettenstandard zeugen nicht gerade von der Qualität eines Personal Computers.

Dies haben auch findige Zubehörlieferanten entdeckt und ein 5 ¼"-Diskettenlaufwerk für den Schneider entwickelt. Nun gibt es auf dem Markt zahlreiche Systeme, die als zweites Laufwerk zur Schneider-Floppy angeschlossen werden. Das hat den Vorteil, daß man nicht auf das Schneider-Format verzichten muß und trotzdem die Leistung eines größeren Laufwerks zur Verfügung hat, obwohl auch hier durch den Schneider Controller die Kapazität auf 40 Spuren pro Diskettenseite beschränkt ist. Müßig ist allerdings die Benutzung von zwei verschiedenen Diskettenformaten – dies würde auch einem professionellen Einsatz als Personal Computer widersprechen.

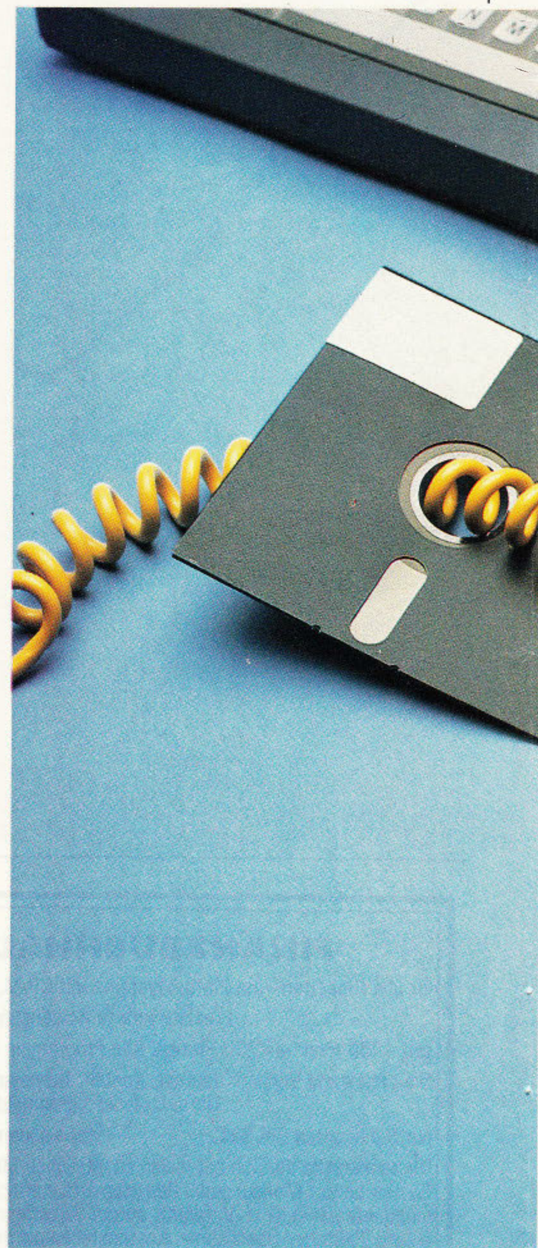
Das hat die Firma Vortex erkannt und eine Diskettenstation mit Controller entwickelt, die durch ihre Leistung für einiges Aufsehen in der Fachwelt gesorgt hat. Von Data Media ist ebenfalls ein 5 ¼"-Laufwerk entwickelt worden, das vom Schneider Controller unabhängig ist. Das Vortex-Laufwerk beschreibt 80 Spuren pro Diskettenseite beidseitig und erhält dadurch eine Kapazität von 704K pro Diskette. Bei einem Doppellaufwerk steht also auf zwei Disketten eine Kapazität von über 1,4 MByte zur Verfügung. Besonders angenehm ist auch die Geschwindigkeit sowie die Gerätegröße, die beim Doppellaufwerk noch nicht einmal das Volumen des Einzellaufwerkes 1541 von Commodore erreicht. Mit diesem Floppylaufwerk wird also der Schneider zu dem, was er in seinem Namen schon vorgibt zu sein, nämlich zum Personal Computer.

CP/M-Diskettenformate

Allerdings nützt das beste Laufwerk recht wenig, wenn keine Software für das verwendete Format zur Verfügung steht. Um diesem Umstand abzuhelfen, hat Vortex ein Programm erstellt, das das BIOS des Laufwerkes programmiert. Hiermit kann menuegesteuert fast jedes Diskettenformat, das ein CP/M-Computer benutzt, auf dem Laufwerk implementiert werden. Dadurch können die Formate der gängigen CP/M-Computer gelesen und beschrieben werden. Wir waren natürlich neugierig, wie das funktionieren würde und beschlossen, von einem OSBORNE das bekannte Textverarbeitungsprogramm Wordstar einzulesen, da wir gerade eine Version für einen OSBORNE zur Verfügung hatten.

Wir legten also die Diskette mit dem Programm PARA, mit dem die Installation der verschiedenen Diskettenformate erfolgen kann, in das obere Laufwerk (Laufwerk A) und haben es geladen. Sofort bekamen wir eine Fehlermeldung, die uns darauf hinwies, daß PARA nur mit einem auf 38K eingestellten CP/M läuft. Dies war aber nicht sehr schlimm, da gleichzeitig auch angezeigt wurde, was wir zu tun hatten. Es war das CP/M-Programm MOVCPM zu laden und damit der Arbeitsspeicher für CP/M einzustellen. Damit dies besser geht, haben wir es direkt auf die PARA-Diskette kopiert und auch die CP/M-Systemspuren mit SYSCOPY auf die PARA-Diskette geschrieben. Nachdem unser CP/M richtig auf 38K eingestellt war, konnten wir mit PARA anstandslos arbeiten.

Das Menue führte uns problemlos zum Ziel. Wir haben Laufwerk B als OSBORNE-Laufwerk installiert. Allerdings fiel uns auf, daß in den bereits vorgegebenen Formaten nur das OSBORNE DD-Format vorhanden war. Da es immer noch viele ältere OSBORNE im SD-Format gibt, wäre es wünschenswert, wenn dies auch gleich mitgeliefert würde. Wir hatten glück-



licherweise eine Wordstar-Diskette im OSBORNE-DD-Format zur Hand. Die Verarbeitung von nicht voreingestellten Formaten ist auch möglich, wenn die einzustellenden Parameter

des gewünschten Formates bekannt sind. Diese sind aber oft schwierig zu erhalten, deshalb ist es sicherer und auch bequemer, mit den voreingestellten Formaten zu arbeiten.

Zur Probe haben wir dann direkt aus dem Programm PARA die Directory der OSBORNE-Diskette gelesen, und das klappte ausgezeichnet. Nach der Rückkehr ins CP/M zeigte uns der Computer an, daß auf Laufwerk A das Vortex-Format und auf Laufwerk B das OSBORNE-DD-Format installiert sei.

Übertragen von Wordstar

Wir haben nun sofort versucht, Wordstar von der OSBORNE-Diskette zu laden. Wie nicht anders zu erwarten, ging das natürlich nicht. Der Computer meldete sich mit der Fehlermeldung, daß der Speicherplatz zu gering sei. Das lag daran, daß wir ja den Speicher im CP/M auf 38K heruntergesetzt hatten, um ein anderes Diskettenformat zu installieren. Wenn wir nun Wordstar auf die Diskette im Laufwerk A, das immer noch auf Vortex-Format eingestellt war, kopieren würden, könnten wir mit einem CP/M arbeiten, das 44K als Arbeitsspeicher zur Verfügung hat.

Wir haben dann Wordstar auf die andere Diskette kopiert, CP/M neu gestartet und dann Wordstar geladen. Das Laden klappte wunderbar, das Programm lief auch, aber der Bildschirm war ziemlich durcheinander geraten.

Dies hatte mit dem uns bekannten Wordstar wenig zu tun. Das ist auch nicht verwunderlich, da der OSBORNE über eine andere Bildschirmansteuerung verfügt als der Schneider. Wie jedes Programm, das bildschirmorientiert arbeitet, muß Wordstar an den benutzten Computer angepaßt werden.



Bildschirmanpassung

Für die notwendige Anpassung haben wir das Programm INSTALL.COM, das sich auf der Wordstar-Diskette befindet, geladen und mit der Installierung begonnen. Für denjenigen, der noch nicht mit diesem Installationsprogramm gearbeitet hat, ist es sicher zunächst etwas verwirrend. Als erstes fragt das Programm – selbstverständlich in englisch – ob Wordstar erstmals eingestellt werden soll. Da wir eine bereits einmal generierte Version hatten – wie alle gängigen Versionen, die erhältlich sind, für einen Computer bereits generiert sind – antworteten wir mit N (nein). Danach erschien eine Auswahlliste, von der uns der Punkt D am besten zusagte. Dort kann ein bestehendes Programm-File geändert und nach Generierung automatisch gestartet werden. Dann fragt das Programm nach der Bildschirmanpassung und stellt eine Anzahl verschiedener Bildschirme zur Auswahl. Der Schneider (oder auch der Amstrad) war, wie wir erwartet hatten, nicht dabei. Deshalb haben wir erst einmal alle angebotenen Terminals der Reihe nach probiert, denn es hätte ja sein können, daß eines paßt und wir uns dann die Arbeit der Generierung "von Hand" hätten sparen können. Wir haben dabei allerlei merkwürdige Bildschirmaufbauten gesehen, die teilweise blinkten und hupten; aber keine der angebotenen Möglichkeiten war so, daß wir damit einverstanden sein konnten.

Letztlich haben wir dann eingegeben, daß keiner dieser Bildschirme vorhanden ist.

Im nächsten Menue mußte der Drucker vorgegeben werden. Wir haben die Option A gewählt, da damit die meisten Drucker laufen. Dann kam die Schnittstelle an die Reihe. Der Schneider verfügt über eine Centronics-Parallel-Schnittstelle, das ist in dem Menue die Auswahl N. Nun mußten wir noch das Kommunikationsprotokoll mit N als nicht notwendig angeben und konnten uns dann durch Eingabe von N auf die Frage, ob die Generierung abgeschlossen sei, an die Bildschirmanpassung begeben. Das geht so, daß zuerst die zu ändernde Speicherstelle in Hexadezimalschreibweise eingegeben wird, dann zeigt der Computer den bisherigen Wert an und fragt nach dem neuen Wert. Die Zahl muß auch hier in HEX eingegeben werden. Mit der Entertaste kann entweder der jeweilige Wert bestätigt oder bei der Speicherstelle um einen Wert erhöht werden.

Wir haben mit der Bildschirmgröße begonnen und dazu muß in 0248 eine 19 und in 0249 eine 50 geschrieben werden. Zur optimalen Cursoranpassung sollten jetzt folgende Werte eingegeben werden: 024A:01, 024B:1F (Cursorpositionierung), 024D bis 025C jeweils 00, 025D:FF (für Spalte vor der Zeile), 025E:01, 025F:01 (Offset auf Spalte und Zeile, das ist notwendig, da die linke obere Ecke des Bildschirmes statt durch



die Koordinaten (0,0) durch (1,1) bezeichnet wird), 0260 bis 0265 jeweils 00, 0266:C9 (unbenutzte Routine mit Rücksprung), 0267 bis 026E jeweils 00, 026D:01,026E:12 (Löschen bis Zeilenende), 0270 bis 0283 jeweils 00. Es kann auch jetzt bis 02C4:00 eingegeben werden, jedoch führt die Eingabe von 0284:01, 0285:18, 028B:01, 028C:18 dazu, daß das Menue von Wordstar invers auf dem Bildschirm angezeigt wird. Hier kann jeder Benutzer nach Belieben für sich entscheiden, was ihm lieber ist. Eine 00 in diesen Stellen führt zur normalen Anzeige.

Jetzt müssen noch bis Speicherstelle 02CE jeweils 00 eingegeben werden, mit Ausnahme der Stellen 02BC, 02BF, 02C2, 02C5 und 02CB, in denen eine C9 für den Rücksprung aus der unbenutzten Routine zu finden sein muß. Wichtig sind jetzt noch folgende Werte, die wahrscheinlich so schon eingestellt sind: 02CF:03, 02D0:09, 02D1:10, 02D2:05, 02D3:05. Mit der Eingabe der Adresse 00 wird die Generierung abgebrochen; das Programm fragt mehrmals, ob die Parameter so gespeichert werden sollen; nach der Bestätigung wird dann Wordstar gestartet und sollte jetzt einen funktionsfähigen Bildschirmaufbau liefern.

Es sollte noch darauf hingewiesen werden, daß diese Anpassungen selbstverständlich auch mit dem Debugger DDT, der zu jeder CP/M-Systemdiskette gehört, vorgenommen werden können. Die Adressen und die Werte ändern sich dadurch nicht. Bei neueren Installationsprogrammen zu Wordstar muß nicht mehr unbedingt die Speicheradresse angegeben werden Hier genügt es oft, bei der Abfrage der Funktion den richtigen Wert einzugeben. Dafür ist in unserer Anleitung die Funktion der jeweiligen Adresse angegeben. Mit einigen weiteren interessanten Wordstaradressen sollten wir uns noch beschäftigen, da sie das Arbeiten erheblich vereinfachen können. In Speicherstelle 360 kann das Hilfsmenue in seiner Grundeinstellung vorgegeben werden. Enthält die Adresse eine 0, erfolgen keine Hinweise von Wordstar mehr, bei einer 1 wird kein Hilfsmenue angezeigt, bei einer 2 werden nur die Untermenues gezeigt, und bei einer 3 werden alle Menues angezeigt. Dies bezieht sich auf die Grundeinstellung und kann im Hilfsmenue J immer noch geändert werden. Für einen geübten Benutzer ist es aber oft sehr lästig, die Menues nach Programmstart erst einmal abschalten zu müssen. Hier kann die Grundeinstellung nach den eigenen Bedürfnissen erfolgen. Den Blocksatz kann man mit Adresse 398 ein- bzw.

ausschalten. Enthält sie eine 0, ist der Blocksatz ausgeschaltet, und bei FF ist er eingeschaltet. Ob das Inhaltsverzeichnis der Diskette standardmäßig angezeigt werden soll, kann in Speicherstelle 363 festgelegt werden.

Die Directory wird bei Programmstart ausgegeben, wenn die Adresse FF enthält, bei 0 wird es unterdrückt. Will man statt im Einfügemodus im Überschreibmodus arbeiten, dann braucht lediglich in Stelle 362 eine 0 enthalten zu sein. Bei einem FF ist der Einfügemodus eingeschaltet.

Druckerprobleme

Nun läßt sich wie gewohnt mit Wordstar arbeiten. Wer allerdings glaubt, er könne jetzt, da dies bei der Installation so eingegeben wurde, einen normalen Centronics-Drucker an den Druckerport anschließen, muß enttäuscht werden. Mit dem Schneider Drucker und dem mitgelieferten Kabel funktioniert das wunderbar. Wer allerdings ein an-



deres Kabel als das Schneider Kabel benutzt, wird feststellen, daß nach jeder gedruckten Zeile eine Leerzeile eingefügt wird. Dies liegt daran, daß beim Schneider der Centronics-Port auf Stift 14 mit Masse belegt ist, was bei fast allen Druckern einen automatischen Zeilenvorschub verursacht. Schneider hat das Problem so gelöst, daß in dem Flachbandkabel die Phase, die den Pin 14 verbindet, durchtrennt wurde. Da wir unser Kabel auch noch für andere Drucker und Computer verwenden wollten, haben wir uns mit einem kleinen Trick geholfen. Mit Hilfe eines Klebestreifens haben wir den Pin 14 im Druckerport abgeklebt, so daß dort keine Verbindung mehr zustande kommen konnte. Der Pin 14 befindet sich im Druckerport von oben gesehen auf der rechten Seite als 3. Stift. Da die beiden rechts daneben liegenden Kontakte nicht belegt sind, haben wir die der Einfachheit halber auch direkt mit abgeklebt. Nach dieser kleinen Operation funktionierte unser Drucker wie gewohnt. Was bei der Druckeranpassung noch fehlt, ist die Einstellung von Wordstar auf die verschiedenen Druckertypen. Da es aber eine Vielzahl von Druckern gibt, soll an dieser Stelle auf nähere Einzelheiten verzichtet werden.

Cursorsteuerung

Durch die Drucker- und Bildschirmanpassung kann mit dem Programm gearbeitet werden, aber noch nicht so komfortabel wie es sein könnte. Wir stellten bei näherer Betrachtung nun fest, daß die Cursortasten alle möglichen Zeichen enthalten, nur nicht die Cursorsteuerung; die ESC-Taste ist auch nicht mit Escape belegt. Es ist zwar möglich, mit Wordstar zu arbeiten, da es Computer gibt, die nicht über Cursortasten verfügen, so daß mit der CTRL-Taste gemeinsam mit verschiedenen Buchstaben der Cursor bewegt werden kann, aber man kann es sich auch besser vorstellen. Unter CP/M ist es, sowohl mit dem von Schneider, als auch mit dem von Vortex vertriebenen System möglich, die Tastatur den eigenen Wünschen entsprechend zu belegen.

Tastaturbelegung

Bei Vortex heißt das für diesen Zweck vorgesehene Programm INSTALL (nicht verwechseln mit dem INSTALL bei Wordstar), bei Schneider heißt es SETUP.COM. Nach dem Aufruf von INSTALL meldet sich das Programm mit einem Menue, das aus fünf Punkten besteht. Nummer 2 beinhaltet die Tastaturbelegung. Nachdem wir die Taste "2" gedrückt haben, gibt uns der Computer folgende Meldung bzw. Auswahlliste:

1 NORM Tabelle ist leer
2 SHIFT Tabelle ist leer
3 CTRL Tabelle ist leer
4 Funktionstasten Tabelle ist leer
5 Hauptmenue

Das bedeutet, wir haben vier Möglichkeiten zum Belegen der Tasten zur Verfügung.

Die drei Ebenen der Tastatur des Schneider sind uns schon von Basic oder anderen Programmen bekannt. Normalerweise befinden sich auf der Tastatur die Kleinbuchstaben. Zusammen mit der SHIFT-Taste kann man die Großbuchstaben erhalten. Wenn in Basic eine Taste gemeinsam mit der CTRL-Taste gedrückt wird, erscheint auf dem Bildschirm ein graphisches Symbol. Standardmäßig stellen diese Symbole Steuerzeichen für die Drucker- und Bildschirmanpassung dar. Im ASCII-Code sind das die Zeichen kleiner als 30. Für die Bildschirmanpassung von Wordstar haben wir diese Zeichen bereits benutzt.

In Basic können die Tasten relativ leicht umdefiniert werden. Mit den Befehlen KEY und KEYDEF ist das unproblematisch. Es gibt auch bereits Programme, die das ausnutzen oder die Tastaturdefinition unterstützen. Ein Beispiel dazu, wie die Zehnertastatur mit wichtigen Basicbefehlen belegt wird, findet sich beispielsweise in dem Buch "CPC 464 für Ein- und Umsteiger" aus dem Verlag Markt & Technik auf Seite 230.

Wordstar-Steuer-Codes

Unter CP/M ist die Tastatur auch mit Steuerzeichen auf der CTRL-Ebene belegt. Wordstar benutzt diese ASCII-Zeichen zur internen Befehlsverwaltung. Um die Tastatur anzupassen, müssen wir erst einmal die wichtigsten Tastencodes in Hexadezimalzahlen herausfinden. Für die Wordstar-Befehle ergeben sich folgende Codes:

Tabelle 1

ESC	1b	
CTRL A	01	Cursor Wort links
CTRL S	13	Cursor Zeichen links
CTRL D	04	Cursor Zeichen rechts
CTRL F	06	Cursor Wort rechts
CTRL E	05	Cursor hoch
CTRL X	18	Cursor runter
DEL	7f	Zeichen links löschen
CTRL G	07	Cursorzeichen löschen
CTRL Z	1a	Zeile rauf rollen
CTRL W	17	Zeile ab rollen
CTRL T	14	Wort löschen
CTRL Y	19	Zeile löschen
CTRL P	10	Druckermenue
CTRL O	0f	Formatierung
CTRL K	0b	Diskettenmenue
CTRL J	0a	Hilfsmenue
CTRL Q	11	Kommandomenue
CTRL I	09	Tab
CTRL N	0e	Leerzeile

Nachdem uns nun diese Steuercodes bekannt sind, können wir die Tastatur so belegen, wie es uns gefällt. Die Mindestanforderung dürfte darin bestehen, daß die Cursorsteuerung über die Cursortasten und das Löschen der Buchstaben mit DEL und CLR erfolgt, daß ESC auch auf ESC zu finden ist und daß TAB auf der TAB-Taste liegt. Bei Betrachtung der Standardbelegung (Schneider Handbuch Anhang III, Seite 14) stellen wir fest, daß DEL und TAB bereits richtig belegt sind. Die sonstige Anpassung müssen wir nun vornehmen.

Dafür drücken wir im INSTALL-Programm unter dem Menuepunkt 2 (Tastaturbelegung) die Taste 1 zum Ändern der NORM-Tabelle. Danach wird die bisherige Normtabelle – sie ist bisher leer – und ein neues Menue zur Änderung der Tabelle auf dem Bildschirm ausgegeben. Wir wählen 3: Neueintrag. Das Programm fragt nun nach der Tastennummer und nach dem Token, das ist der Code des Befehls. Die Tastaturnummer finden wir im Schneider Handbuch auf Seite 16 im Anhang III. Die ESC-Taste hat die Nummer 66.

Wir geben also bei der Frage nach der Taste die Nummer 66 ein. Diese darf in Dezimalschreibweise, also so, wie sie im Handbuch steht, eingegeben werden.

Auf die Frage nach dem Token tippen wir die Hexadezimalzahl 1b ein, was nach unserer Tabelle der Befehlscode für ESC ist. Danach erhalten wir die neue Normtabelle, die jetzt einen Eintrag enthält und müssen erneut die Taste 3 im Menue drücken, um einen weiteren Eintrag vorzunehmen. In dieser Weise speichern wir für CLR die Tastennummer 16 und Token 07. Die Cursorsteuerung geben wir folgendermaßen ein:

Tabelle 2

Taste	Token	
0	05	(Cursor rauf)
8	13	(Cursor links)
1	04	(Cursor rechts)
2	18	(Cursor runter)

Damit ist unsere Grundeinstellung



komplett. Der Cursor läßt sich über die Cursortasten steuern, ESC, TAB und das Löschen der Zeichen geschieht mit den richtigen Tasten. Nun können wir uns noch einige Vereinfachungen eingeben. Beim Formatieren mit CTRL B fragt Wordstar immer wieder Trennungen ab. Der Cursor kann dabei noch bewegt werden und der Trennstrich (-) soll an der richtigen Stelle eingegeben werden. Praktisch ist es, wenn der Trennstrich sich auf der COPY-Taste bei den Cursor-Tasten befindet, da dann mit den Tasten Cursor rechts und Cursor links die richtige Stelle gewählt werden kann und mit der dazwischen liegenden COPY-Taste die Trennung gesetzt wird. COPY hat die Tastennummer 9 und der Bindestrich Token 2d.

Damit wollen wir die Normtabelle abschließen.

Es ist sinnvoll, die Cursortasten noch auf der Shift- und CTRL-Ebene zu belegen. Auf der Shift-Ebene können wir das Rollen des Bildschirmes auf und ab mit Cursor hoch und Cursor runter erreichen, sowie das Verschieben des Cursors wortweise mit Cursor rechts und Cursor links. Dazu wählen wir im INSTALL-Programm die Veränderung der Shift-Tabelle an und geben folgende Werte ein:

Tabelle 3

Taste	Token	
0	1a	(Rolle rauf)
8	13	(Wort links)
1	04	(Wort rechts)
2	17	(Rolle ab)

Funktionstasten

Nun ist unser Wordstar schon recht komfortabel. Aber es geht noch besser.

Wir wissen, daß das Ende der Bearbeitung mit der Tastenfolge CTRL K und anschließend D geschieht. Schön wäre es, diese Tastenfolge auf eine Taste (am besten die Shift-COPY-Taste) zu legen. Um das zu erreichen, muß ein Token definiert werden, das diese Tastenfolge enthält. Im INSTALL-Programm geschieht dies über die Funktionstasten-Tabelle (Menuepunkt 4).

Hier wird erst nach der zu definierenden Tokennummer gefragt. Diese muß im Bereich zwischen 80 und 9f (jeweils hexadezimal) liegen. Die erste Funktionstaste (CTRL K, D) legen wir auf Token 80. Hier können wir wählen, ob eine Zeichenfolge in HEX-Zahlen oder in Buchstaben eingegeben werden soll.

Mit CTRL H wird auf HEX-Eingabe und mit CTRL C auf Eingabe normaler Buchstaben (ASCII) umgeschaltet.

Wir wollen mit dem Code für CTRL K beginnen und geben nach entsprechender Umschaltung den HEX-Wert 0b ein. Danach schalten wir auf ASCII-Eingabe um und drücken die Taste D.

Mit der Enter-Taste beenden wir die Eingabe und können nun die nächste Token-Nummer in analoger Weise belegen. Es scheint zweckmäßig zu sein, auch noch das Springen zum Textanfang und zum Textende vorzuprogrammieren. Das sind CTRL Q,R und CTRL Q,C. Wir belegen mit dieser Tastenfolge (CTRL Q hat den HEX-Wert 11) die Token 81 und 82. Wenn wir nun in den anderen Menuepunkten (NORM-, SHIFT-, CTRL-Tabelle) eines dieser Token auf eine Taste legen, so wird diese Taste zur Funktionstaste. Legen wir also in der Shift-Tabelle Token 80 auf Taste 9, können wir ab sofort in Wordstar mit Drücken der Shift-Copy-Taste die Bearbeitung beenden und den Text speichern. In der CTRL-Ebene können wir jetzt die Token 81 und 82 auf die Tasten 0 und 2 legen und mit CTRL Cursor rauf zum Textanfang oder mit CTRL Cursor runter zum Textende springen. Auf diese Art kann die gesamte Tastatur zu Funktionstasten umbelegt werden.

Man könnte nun noch CTRL Cursor rechts mit Springen zum Zeilenende und CTRL Cursor links mit Springen zum Zeilenanfang belegen. Die Textformatierung oder die Druckersteuerung kann man auf die Zehnertastatur rechts neben der Normaltastatur legen.

Jede gewünschte Kombination ist möglich und nur durch den Speicherplatz von maximal 128 Zeichen für die Belegung der Funktionstasten begrenzt.

Auch ist es denkbar, ganze Textketten, wie die Anrede in einem Brief (Sehr geehrter) oder den Schluß (Mit freundlichen Grüßen), auf eine Taste zu legen und sich damit das Briefeschreiben erleichtern.

Unterschiede zwischen INSTALL und SETUP

Die Tastaturbelegung ist nach Generierung durch INSTALL vom Programm unabhängig im CP/M enthalten und kann in der gleichen Weise für jedes andere Programm verwendet werden. Hat man also einmal die Tastatur an Wordstar optimal angepaßt, kann damit auch ein Wordstar-ähnlicher Editor, wie beispielsweise Turbo-Pascal, mit dieser angepaßten Tastatur bedient werden.

Es wurde bereits erwähnt, daß diese Tastaturanpassung sowohl mit dem Programm INSTALL von Vortex, als auch mit SETUP von Schneider erfolgen kann. Beim SETUP ist aber auf eine Besonderheit hinzuweisen. Das Programm unterscheidet nicht – wie oben für INSTALL beschrieben – die Normal-, Shift- und CTRL-Ebene in verschiedenen Menues, sondern erwartet die Definition einer Taste auf allen ihren Ebenen. Die Eingabe erfolgt durch Tastennummer, Token Normalebene, Token Shiftebene, Token CTRL-Ebene. Man muß also vorher die Belegung etwas besser planen als bei INSTALL, weil man gleichzeitig sozusagen in drei Ebenen denken muß. Dieser Unterschied bezieht sich aber nur auf die Bedienung, nicht auf die Funktion. Die Belegung der Tastatur nach eigenen Wünschen und die Definition von Funktionstasten ist mit SETUP in ähnlicher Weise wie bei INSTALL möglich.

Wordstar läuft nach diesen Anpassungen auf dem Schneider mindestens so gut wie auf jedem anderen CP/M-System, vielleicht sogar durch die Belegung von Cursor-Tasten und Zehnerblock noch bedienungsfreundlicher als bei manchem System, das nur eine Normaltastatur hat. Es ist also möglich, ein Wordstar, das im Büro auf einem professionellen Computersystem läuft, auch auf den Schneider zu übertragen und damit zu Hause weiterzuarbeiten.

Wer also bereits eine Wordstar-Version hat, braucht nicht unbedingt für den privaten Bereich ein neues Textverarbeitungsprogramm anzuschaffen und zu erlernen. Allerdings müssen wir erkennen, daß die Funktion "Verschieben von Blöcken", aufgrund Speicherplatzmangels, nur mit einem sehr kleinen Block durchgeführt werden kann. Dieses Problem kann aber umgangen werden, wenn der zu verschiebende Block auf der Diskette, die beim Vortex-Laufwerk mit 704K in jedem Falle genügend Kapazität bietet, zwischengespeichert und an der richtigen Stelle wieder eingeladen wird. Damit dies nicht erforderlich ist und Wordstar, sowie jedes andere CP/M-Programm, auf dem Schneider anstandslos läuft, haben Vortex und Data Media eine Speichererweiterung entwickelt. Damit dürfte der Schneider dann den letzten Schritt zum Personal-Computer vollzogen haben.

Anwendung für SUBMIT

Die Qualität eines Betriebssystems zeigt sich erst, wenn man mehrere Be-



fehlsfolgen miteinander verknüpfen will. In Basic erfolgt das jeweils durch ein Programm, indem die Befehle enthalten sind. Sollen mehrere Programme hintereinander abgearbeitet werden, ist das unter Basic, obwohl das Schneider Basic die notwendigen Befehle besitzt, nur schwer erreichbar. Jedes vorhergehende Programm muß die Laderoutine für das Folgeprogramm enthalten. Veränderungen der Reihenfolge oder auch die Hinzunahme oder das Wegfallen einzelner Programme wird damit fast unmöglich. Bei CP/M gibt es ein Programm auf der Systemdiskette, mit dem eine solche Verknüpfung leicht möglich ist. Das Programm heißt SUBMIT und teilt CP/M mit, welcher Befehl bzw. welches Programm nun abgearbeitet werden muß.

Dazu legt es eine Systemdatei an, in der die erforderlichen Hinweise für das Betriebssystem enthalten sind.

Ein Anwendungsbeispiel ergibt sich in Verbindung mit Wordstar, besonders bei einem Wordstar, das von einem anderen Computer angepaßt worden ist und deshalb keinen deutschen Zeichensatz enthält. Von der Firma Schneider Data (früher Escon) gibt es ein Programm, das eine deutsche Tastatur mit deutschem Zeichensatz unter CP/M herstellt. Nun könnte, nachdem CP/M gestartet ist, erst die Tastaturdefinition geladen und dann das Laufwerk auf B umgestellt werden, damit Wordstar die Datendiskette in Laufwerk B benutzt, und danach könnte dann Wordstar von A geladen werden. Wir erstellen dazu eine Datei, die im Namen hinter dem Punkt SUB enthalten muß. Die Datei kann mit Wordstar – oder jedem anderen Texteditor, z.B. ED – erstellt werden. Wir nennen unsere Datei WS-LOAD.SUB, da wir damit die Laderoutine von Wordstar erstellen wollen. Die Datei müßte nun folgende Einträge enthalten:

```
DEUTSCH  <
B:       <
A:WS     <
```

Mit dem Befehl DEUTSCH soll das Programm mit der deutschen Tastatur geladen werden. Es muß dazu natürlich unter dem Namen DEUTSCH.COM auf der Diskette enthalten sein. Mit B: wird auf Laufwerk B umgeschaltet und mit A:WS soll Wordstar von der Diskette in Laufwerk A geladen werden. Das < soll andeuten, daß jede Zeile mit der Enter-Taste abgeschlossen werden muß.

Für die Besitzer einer Vortex-Diskettenstation mit dem Diskettenbetriebssystem VDOS 2.0 ergibt sich noch eine weitere Möglichkeit. Man kann hier auswählen, ob der Bildschirmaufbau in einer schnelleren Routine erfolgen soll, bei der die Windows nicht richtig benutzt werden können, oder ob er im normalen Tempo erfolgen soll. Da Wordstar die Windowtechnik nicht benutzt, bietet sich der schnellere Bildschirmaufbau an. Die SUB-Datei sieht dann folgendermaßen aus:

```
DEUTSCH  <
FAST     <
B:       <
A:WS     <
```

Wir wollen nun, daß diese Befehlsfolge sofort abgearbeitet wird, wenn CP/M gestartet wird. Dafür gibt es im Betriebssystem den Autostart-Bereich. Er kann mit dem INSTALL- oder SETUP-Programm definiert werden. Über Menue ist der Autostartstring, so heißt das zu startende Programm in INSTALL, sofort aufrufbar. In unserem Fall muß der Autostartstring heißen: SUBMIT WSLOAD. Er muß noch mit einem Carriage Return versehen werden, das ist der ASCII-Code 13 oder in hexadezimaler Schreibweise 0D. Der Effekt ist, daß nach Starten von CP/M SUBMIT automatisch gestartet wird und die Befehle, die in der Datei WSLOAD.SUB enthalten sind, der Reihe nach abarbeitet.

Für die Vortex-Besitzer mit VDOS 2.0 bietet sich hier sofort die Möglichkeit an, mit einem Programm, das HELLO.-BAS heißt und auf der Wordstardiskette enthalten ist, allein durch Einschalten von Laufwerk und Computer (oder durch Reset) automatisch CP/M zu laden und dann durch Autostart und SUBMIT die obige Befehlsfolge abarbeiten zu lassen. Das Programm ist sehr einfach und sieht folgendermaßen aus:

10 CP/M

Es genügt also lediglich das Einlegen der richtigen Diskette in Laufwerk A und Anschalten des Computers, um in wenigen Sekunden mit Wordstar arbeiten zu können. Bequemer geht es eigentlich nicht mehr. Mit einer solchen Verknüpfung lassen sich auch noch weitere positive Effekte erzielen. Oft ist es erwünscht, nach der Bearbeitung durch Wordstar, die Datendiskette zu kopieren. Bei Verwendung von SUBMIT kann unsere Liste leicht durch ein Kopierprogramm wie PIP oder FILECOPY ergänzt werden. Dann sieht unsere Datei WSLOAD.SUB folgendermaßen aus:

```
DEUTSCH   <
FAST      <
B:        <
A:WS      <
A:        <
FILECOPY  <
```

Wird nun Wordstar durch Eingabe von X im Grundmenue beendet oder die Dateibearbeitung durch CTRL K,X abgebrochen, wird nicht sofort ins Betriebssystem zurückgesprungen, sondern das Kopierprogramm FILECOPY geladen. Es muß selbstverständlich auf der Wordstardiskette enthalten sein. So kann bequem und schnell eine Sicherheitskopie angefertigt werden, ohne daß jedesmal das notwendige Programm geladen werden muß. Es könnte dort auch COPY stehen, wenn die Diskette jedesmal ganz kopiert werden soll.

Erst durch die Verwendung von SUBMIT wird die Leistungsstärke von CP/M voll ausgenutzt. Die Bedienungsfreundlichkeit, die man von einem Personalcomputer erwartet, kann hier geboten werden. Jeder, der darauf verzichtet, verzichtet auf ein Stück Komfort und auch auf Professionalität. Mit unserer Wordstaranpassung, Tastaturanpassung und der Laderoutine mit SUBMIT wird die Leistung des Schneiders tatsächlich mit einem Personalcomputer vergleichbar.

Hartmut Pick

Tabelle der einzugebenden Codes für die Anpassung Wordstar an Schneider:

0248 19 Bildschirmhöhe
0249 50 Bildschirmbreite
024A 01
024B 1F Cursorpositionierung
024D bis
025C 00 unbenutzt
025D FF für Spalte vor der Zeile
025E 01
025F 01 Offset auf Spalte und Zeile
0260 bis
0265 00
0266 C9 unbenutzte Routine mit Rücksprung
0267 bis
026E 00
026D 01
026E 12 Löschen bis Zeilenende
0270 bis
0266 C9 unbenutzte Routine mit
02C5 bis Rücksprung, müssen alle
02BB 00 auf 00 gesetzt werden
02BC C9 und Rücksprung auf C9
02BD bis
02BE 00
02BF C9
02C1 00
02C2 C9
02C3 bis
02C4 00
02C5 C9
02C6 bis
02CA 00
02CB C9
02CF 03 Grundeinstellung
02D0 09 in Wordstar, sehr
02D1 10 wichtig!
02D2 05
02D3 05



Ein alter Standard lebt wieder auf: CP/M. Dies war wohl nicht zuletzt auch deshalb möglich, weil mit Computern, wie den Schneider CPC's, preisgünstige Rechner auf den Markt kamen, die mit Einschränkungen CP/M-fähig sind. Hieß es anfangs zwar noch, CP/M gut und schön, aber wie solle man denn bitteschön mit diesem unmöglichen Diskettenformat des Schneiders an die Software kommen, ohne die natürlich auch das beste Betriebssystem nichts bringt, so muß man heute diesen Einwand entkräften: Es besteht nun ja auch die Möglichkeit 5 ¼"-Laufwerke von Fremdherstellern an die CPC's anzuschließen.

WordStar und DDT

Wordstar auf dem Schneider: Wohin mit dem Text?

Außerdem sind immer mehr Softwarehäuser dazu übergegangen, alte CP/M-Programme auch auf 3"-Disketten anzubieten, und dies zu Preisen, wie sie vor zwei Jahren noch unvorstellbar gewesen wären. Das hat aber seinen guten Grund, denn diese Programme haben sich natürlich längst amortisiert und können nun, anstatt irgendwo zu verstauben, den Herstellern noch einmal kräftig etwas einbringen. Insbesondere die Preispolitik des Markt & Technik Verlages ist zu erwähnen. Hier werden die Programme WordStar, dBase II und Multiplan nun nach Monaten noch zum Einführungspreis von knapp 200,– DM verkauft. Zwischendurch stieg der Preis zwar kurz auf 300,– DM, aber man hat wohl eingesehen, daß damit die Schmerzgrenze für viele Anwender überschritten war, zumal man natürlich weit unter diesem Preis andere Textprogramme erhalten kann, die ähnliche Leistungsmerkmale wie der legendäre WordStar aufweisen können, und dabei sogar noch besser an die Schneider Computer angepaßt sind.

Nennen ließe sich hier beispielsweise der Star-Writer I von Star Division. Sind nun auch die Diskettenformatprobleme, zum Teil wenigstens, überwunden, so tauchen neue auf. CP/M-Programme gehen sehr freizügig mit dem Speicherplatz auf der Diskette um, und die CPC's bieten nun mal nicht allzuviel davon. Außerdem brauchen dBase und Multiplan auch eine größere TPA (Transient Program Area), als ein CPC (insbesondere die der »64« Reihe) sie bieten kann. Mit WordStar läßt sich zwar arbeiten, will man ihn aber nicht nur zum gelegentlichen Briefeschreiben benutzen, so muß man bedenken, daß es bei den Kosten für das Programm allein nicht bleiben wird, denn es werden alsbald Hardware-Erweiterungen notwendig.

Eine 3"-Diskette bietet unter CP/M ungefähr 169KB Speichervermögen. Davon belegt WordStar (ohne MailMerge, das man ohnehin wohl nur selten benutzt) gut 76KB. Hat man nun noch 2 – 3 Dienstprogramme auf der Diskette, sind gut und gerne 90KB einfach weg. Auch die nun verbleibenden 79KB können nicht für den Text allein genutzt werden. Man muß sie vielmehr noch einmal dritteln, wenn man ordentlich arbeiten will. Das andere Drittel frißt die Zwischendatei, die WordStar während der Texterstellung anlegt.

Bleiben für den eigentlichen Text also – sagen wir mal – 25KB. Das ist verflixt wenig. Folglich muß ein zweites Laufwerk her oder eine Speichererweiterung, die eine RAM-Disk zur Verfügung stellt.

Ich rate zu Letzterem, denn das Arbeiten mit WordStar, der ständig auf die Diskette zugreifen muß, ist nur unter einer RAM-Disk ein wahres Vergnügen.

Nun erst, egal ob mit zweitem Laufwerk oder RAM-Disk, bietet sich die Möglichkeit, eine Programm- und eine Datendiskette anzulegen. So wäre es zumindest wünschenswert. Hat jedoch auch das zweite Laufwerk keine allzugroße Kapazität, so gibt es einen besseren Trick, indem man

ws textname.ext d:

eingibt, wobei d: für das zweite Laufwerkt steht. Nur so kann der Speicherplatz der Diskette optimal ausgenutzt werden. WordStar wechselt nun jedesmal beim Abspeichern des Textes die Diskette, d.h., einmal ist der Text jetzt auf Laufwerk A und die Backup-Datei auf Laufwerk B, aber nach dem nächsten Abspeichern ist es schon gerade umgekehrt. So wird es möglich, Texte der Länge zu erstellen, wie sie die Diskette bietet, zieht man die WordStar Dateien ab.

Will man seinen WordStar nun in Laufwerk B oder auf einer RAM-Disk installieren, so kann es leicht passieren, daß das Programm aussteigt, denn WordStar sucht seine Overlay-Dateien immer auf Laufwerk A. Gerade aber bei einer RAM-Disk wäre es doch von Vorteil, einfach mit

pip c:=a:ws*.*

den ganzen WordStar auf der RAM-Disk (also im internen Speicher des Rechners) zu haben, um damit die Zugriffsgeschwindigkeit zu optimieren.

Dazu aber muß dies WordStar erst einmal mitgeteilt werden, daß er zukünftig seine Overlays auf C (oder B) zu suchen hat. Dies ist nicht weiter schwer. Entweder installiert man WordStar neu oder man patcht ihn mit Hilfe des auf der CP/M-Diskette mitgelieferten Debug-Programms DDT. In diesem Fall lautet die Patchadresse 02DC, in die man die Laufwerksnummer zu schreiben hat, also 01 für A, 02 für B, 03 für C usw.

Doch bevor wir nun die einzelnen Schritte genau erläutern, ist es an der Zeit, ein paar Worte über dieses Debug-Programm zu verlieren, denn nicht jeder kennt sich damit aus. Beherrscht man es aber, hat man ein überaus leistungsfähiges Werkzeug zur Verfügung.

DDT (Dynamic Debugging Tool) ist ein Programm, das bei der Fehlersuche helfen soll. Doch es kann nicht nur dem Programmierer nutzen, sondern auch dem, der eigentlich nur Anwendungssoftware laufen lassen möchte. So läßt sich beispielsweise der oben erwähnte Patch sehr gut mit Hilfe von DDT vollziehen.

Es gibt zwei Arten, DDT zu starten: Entweder man ruft es einfach nur auf, oder man gibt noch den Namen eines Programmes an, das untersucht werden soll, also

ddt ws.com

So wird erst DDT in den Hauptspeicher geladen. Dies wiederum lädt dann ws.com in den Bereich ab 0100H. Wenn sich DDT mit seinem Prompt (-) meldet, stehen verschiedene Kommandos zur Verfügung, die einem behilflich sein können. Wollen wir also die grundsätzliche Frage klären, wie sich ein Programm, wie beispielsweise WordStar, mit Hilfe von DDT patchen läßt. Kommen wir auf unser Anfangsvorhaben zurück: Wir wollten WordStar so installieren, daß er seine Overlay-Dateien auf einem anderen Laufwerk sucht. Dazu geben wir wie oben **ddt ws.com** ein, worauf sich der Programmzähler und dann der Prompt meldet. Da unsere Patchadresse feststeht, schreiben wir hinter den Prompt:

S <patchadresse>,

wobei S für den Befehl **Set memory to value** steht. Zwischen dem Befehl S und der Patchadresse darf kein Leerraum sein. In unserem Fall lautet die Eingabe also:
S02DC



DDT zeigt daraufhin die entsprechende Speicherstelle an, dahinter ihren derzeitigen Inhalt. War WordStar bisher so installiert, daß er seine Hilfsdateien auf A suchte, so zeigt DDT an

02DC 01_

Der Cursor weist uns darauf hin, daß nun eine Eingabe von uns erwartet wird. Wollen wir WordStar nun auf B installieren, geben wir 02, für Laufwerk C 03 usw., ein. Soll die betreffende Speicherstelle vollkommen unverändert bleiben, so genügt ein <ENTER>, woraufhin die nächste Speicheradresse angezeigt wird. Ist alles so geändert, wie wir das wollten, geben wir einfach einen Punkt "." ein, woraufhin DDT wieder zu seinem Prompt zurückkehrt. Damit haben wir den WordStar auch schon gepatcht. Aber das leider nur im Inneren unseres Rechners! Irgendwie müssen wir es nun auch noch auf die Diskette schreiben. Dazu geben wir hinter dem Prompt **g0** ein.

```
A>ddt
DDT VERS 2.2
-d7840
7840 00 00 00 00 00 00 00 00 00 53 EF A0 20 6D 61 6E .........S.. man
7850 63 68 65 F2 A0 20 54 65 78 74 AC A0 20 76 6F EE che.. Text.. vo.
7860 20 64 65 ED 20 6D 61 6E 20 73 63 68 6F 6E 20 67  de. man schon g
7870 6C 61 75 62 74 65 AC A0 20 57 6F 72 64 53 74 61 laubte.. WordSta
7880 F2 A0 20 68 61 62 E5 A0 20 69 68 EE 20 8D 0A 76 .. hab.. ih. ..v
7890 65 72 73 63 68 6C 75 63 6B 74 AC A0 20 6C 7B 7E erschluckt.. läß
78A0 F4 20 73 69 63 E8 20 77 69 65 64 65 72 66 69 6E . sic. wiederfin
78B0 64 65 6E AC 20 77 65 6E EE 20 6D 61 EE 20 6E 75 den. wen. ma. nu
78C0 F2 20 65 69 EE 20 62 69 7E 63 68 65 EE 20 69 EE . ei. bißche. i.
78D0 20 8D 0A 64 65 6E 20 45 69 6E 67 65 77 65 69 64  ..den Eingeweid
78E0 65 6E 20 73 65 69 6E 65 73 20 52 65 63 68 6E 65 en seines Rechne
78F0 72 73 20 68 65 72 75 6D 73 74 6F 63 68 65 72 74 rs herumstochert
-m7849,7902,0100
-g0

A>save 2 text
A>type text
So  mancher  Text,  von dem man schon glaubte,  WordStar  habe  ihn
verschluckt,  läßt sich wiederfinden, wenn man nur ein bißchen in
den Eingeweiden seines Rechners herumstochert.
```

Dieser Befehl löst einen Warmstart aus, ohne den Speicherinhalt dabei zu löschen. Das Weitere ist ganz einfach, da CP/M uns mit SAVE einen Befehl zur Verfügung stellt, um den Kernspeicherinhalt auf Diskette zu schreiben. Die Syntax des Befehls lautet:

save länge name.ext

wobei für die Länge jeweils das Vierfache der zu schreibenden KBs einzusetzen ist. Da wir wissen, daß WordStar 16KB groß ist, setzen wir einfach 16*4, also 64 ein:

save 64 wspatch.com

Es empfiehlt sich, vorsichtshalber statt ws.com einen neuen Namen zu wählen, um erst einmal zu testen, ob unser Patch erfolgreich verlaufen ist. Hinterher können wir wspatch.com immer noch mit Hilfe von **REN** in ws.com umbenennen. Wenn man sich hinsichtlich der Länge nicht ganz sicher ist, kann man ohne weiteres auch eine größere Zahl eingeben. Doch gerade bei den Schneiderlaufwerken empfiehlt sich dieses Vorgehen nicht, da dadurch die entsprechende Datei auch mitwächst. Bei den Schneider Floppys aber lohnt es sich, und das wird mir jeder Anwender leidvoll bestätigen können, hartnäckig um jedes Byte zu kämpfen.
Dies oben beschriebene Verfahren ist der einfachste Weg, ein bestimmtes Anwenderprogramm nach eigenen Wünschen abzuändern.
Hat man nun seinen WordStar auf der RAM-Disk und die Speicherkarte bietet genügend Platz, so kann man natürlich auch seinen Text hier noch unterbringen. Dann geht freilich alles wahnsinnig flink, denn beides, Text und Programm, sind nun ja nicht mehr extern, sondern direkt im Rechner gespeichert. Doch sollte man nun nach Beendigung der Arbeit nicht vergessen, den Text auf die tatsächliche Diskette zu schreiben, sonst wäre er unwiderruflich verloren, nachdem man den Rechner erst einmal ausgeschaltet hat. Besser ist es noch, ca. alle 3 – 4 Seiten mit dem ^KO-Befehl den Text auf Laufwerk A zu kopieren. Hält man sich an diese Regel, dann kann selbst ein plötzlicher Stromausfall uns nicht mehr viel anhaben. Aber noch ein kurzer Tip zum Einsatz von Speicherkarten: Oft ist ein Drucker-Spooler eingebaut, also ein Zwischenspeicher, welcher den Text, den man ausdrucken will, vorübergehend aufnimmt, damit der Computer auch während des Druckens für andere Aufgaben bereitsteht. Doch beim Einsatz von WordStar sollte man diesen, falls möglich, ausschalten. WordStar verfügt über einen eigenen Spooler, was zu Konflikten führen kann. Erstens ist es dann nicht so einfach möglich, den Druckvorgang zu stoppen, um beispielsweise einen entdeckten Fehler zu korrigieren; außerdem kann es zu seltsamen, nie eingegebenen Buchstaben am Anfang jeder Seite kommen, so zumindest beobachtet beim Einsatz der Vortex-Speicherkarte. Diese Mißstände lassen sich aber relativ problemlos beheben, indem man den Spooler einfach ausschaltet.

Wie steigt man wieder ein, wenn WordStar aussteigt?

Ein altes, immer neu erlebtes Problem: Man wechselt die Diskette, um einen Block in den laufenden Text einzulesen, arbeitet seelenruhig weiter, bis man es schließlich beim Abspeichern merkt: **Man hat ja vollkommen vergessen, die neue Diskette anzumelden!** Und WordStar kennt in diesem Punkt keine Nachsicht, er steigt aus: Das Betriebssystem meldet sich nun. Es sieht ganz so aus, als wäre alle Mühe vergebens gewesen: Der Text ist weg! Aber jetzt nur keine Aufregung und keinesfalls WordStar etwa neu starten! Ein paar falsche Handgriffe und der Text ist wirklich weg. Als erstes legen Sie eine Systemdiskette, auf der sich auch DDT befindet, in Laufwerk A. Mit Hilfe dieses Programms läßt sich der Text nämlich wieder herstellen. Jetzt melden Sie diese neue Diskette an und starten DDT. Denn dies überaus vielseitige Werkzeug kann uns auch hier helfen. Diesmal rufen wir DDT allerdings ohne einen Zusatz auf, denn wir wollen uns ja nicht irgendein Programm betrachten, um es anschließend unseren Wünschen entsprechend abzuändern, sondern uns vielmehr einmal genauer den Kernspeicherinhalt unseres Computers ansehen. Irgendwo wird der Text doch wohl zu finden sein. Dazu müssen wir das D-Kommando benutzen. **D** steht für **Display memory** und erlaubt uns, verschiedene Speicherbereiche am Monitor sichtbar zu machen. Ab Adresse 0100H ist zumeist das Anwenderprogramm gespeichert, in unserem Fall also DDT selber, da wir es ohne jeden Zusatz aufgerufen haben. Der Text muß also an einer weit höheren Speicherstelle zu finden sein. Stöbern Sie ruhig ein bißchen im Speicher Ihres CPC herum. Solange Sie nur den D-Befehl



gebrauchen kann dabei nichts passieren. Sie brauchen also keinerlei Angst zu haben, nun letztendlich doch noch Ihrem wertvollen Text den Garaus zu machen. Wahrscheinlich werden Sie irgendwo unterhalb der Speicheradresse 8000H fündig. Merken Sie sich die Anfangsadresse Ihres Textes und durchsuchen Sie dann den Speicher weiter, bis Sie auch das Ende gefunden haben. Die Endadresse notieren Sie sich am besten auch irgendwo. Denn DDT kann leider nicht, was viele andere, komfortablere Debugger (wie etwa seine 16-Bit Entsprechung DDT-86) können, nämlich Speicherinhalte direkt auf Diskette schreiben. Ein solches Kommando fehlt unserem Programm einfach.
Sollten Sie also einen Debugger besitzen, der über ein solches Kommando verfügt, müssen Sie den Bereich Anfangsadresse bis Endadresse des Textes einfach unter irgendeinem Namen auf die Diskette schreiben, und schon haben Sie Ihren verlorengeglaubten Text wieder. Doch auch Anwender von DDT brauchen nicht zu verzweifeln. Erinnern wir uns, daß wir vorhin bei dem kleinen WordStar-Patch das residente **SAVE**-Kommando benutzt haben, das es uns ermöglicht, Speicherbereiche ab Adresse 0100H unter Angabe der Länge auf die Diskette zu schreiben. Genau diesen Befehl benötigen wir jetzt. Aber verlassen Sie DDT nicht gleich, denn noch steht unser Text ja in einem vollkommen anderen Speicherbereich. Ab Adresse 0100H aufwärts befindet sich immer noch DDT.COM. Doch dafür hält uns das Programm einen anderen Befehl bereit:

M Anfangsadresse>, Endadresse>, <Zieladresse>

M ist die Abkürzung für **Move Memory** und gestattet uns, beliebig Speicherbereiche zu verschieben. Also genau das, was wir brauchen. Für die Zieladresse müssen wir nur noch 0100 einsetzen, und schon befindet sich unser ganzer Text ab Adresse 0100H. Nun kann man DDT beruhigt, wie oben beschrieben (also durch Eingabe von g0), verlassen und den Text mit SAVE auf Diskette zurückschreiben. Dieses Kommando beherrschen Sie ja inzwischen. Sollten Sie sich über die Länge Ihres Textes nicht ganz im Klaren sein, so können Sie in diesem Fall ruhig eine große Zahl eingeben. Dies ist immer noch besser, als wenn Ihnen sonst genau der wichtigste Teil Ihres Textes fehlen würde. Ist die Textdatei dann erst wieder auf Diskette, können Sie sie ganz normal mit Ihrem WordStar weiterverarbeiten. Aber keine Angst, irgendwann werden Sie wieder einmal vergessen, einen Diskettenwechsel anzumelden. Doch jetzt brauchen Sie sich deshalb keine grauen Haare mehr wachsen zu lassen.

Eine kleine Anregung zum Schluß

Wer hat wohl nicht schon einmal den Wunsch verspürt, den Trennalgorhythmus von WordStar etwas durcheinander zu bringen? Denn da dieser auf amerikanische Verhältnisse zugeschnitten ist, genügt er unseren Ansprüchen nicht immer. Insbesondere Trennmöglichkeiten eines Wortes vor der vierten Stelle übersieht WordStar konsequent. Diese Beharrlichkeit des WordStars muß aber nicht hingenommen werden. Selbstverständlich kann man ihn auch in diesem Punkt abändern. Immerhin ist er ja dafür bekannt, daß man eigentlich alles mit ihm machen kann. Man muß dazu nur wissen, daß der an Speicheradresse 039AH die Information abgelegt ist, ab welcher Stelle in einem Wort WordStar frühestens eine Trennmöglichkeit vorschlägt. Diesen Wert können Sie natürlich verändern. Sie wissen jetzt ja ganz genau, wie man WordStar patcht.

Jürgen Weiß



Tips zu Wordstar

Nachdem bereits in der Schneider International Nr. 1/86 ein Verfahren zur Anpassung des NLQ-Druckers unter Wordstar vorgestellt wurde, sollen hier neben einer weiteren, sehr einfachen Möglichkeit der Druckeranpassung auch noch andere nützliche Tips zum Arbeiten mit Wordstar gegeben werden.

Die Wordstar Versionen ab 3.0

Wordstar ist schon ein etwas älteres Textverarbeitungsprogramm und wurde daher im Laufe der Zeit immer wieder verbessert. Die neueste Version 3.4, wird neben der Version 3.3 am häufigsten unter MSDOS-Systemen angetroffen. Auf den Schneider CPC's läuft in aller Regel die Version 3.0, die vor allem einen gewichtigen Vorteil hat: Sie kann auch ohne die Speichererweiterung von Vortex betrieben werden, wenn auch nur mit einigen Abstrichen.

Auch das von Markt & Technik für die CPC's angebotene Wordstar hat die Versions-Nr. 3.0. Dabei wurde nicht nur die Bildschirmsteuerung angepaßt, sondern auch die Cursorsteuerung des CPC's unterstützt. Um die Ansteuerung des NLQ-Modus beim Schneider Drukker muß man sich allerdings selber kümmern.

NLQ mit Wordstar

Das läßt sich zum einen über das mitgelieferte Installationsprogramm unter Anleitung des bereits erwähnten Artikels in der Schneider International Nr. 1/86 bewerkstelligen. Eine weitere, sehr einfache Möglichkeit besteht darin, Texte unter Wordstar nur mit NLQ und 1 1/2 Zeilen Abstand auszudrucken. Dazu dient das folgende kleine Basic-Programm:

```
10 PRINT #8,CHR$(27);"I";CHR$(3);
20 PRINT #8,CHR$(27);"3";CHR$(51);
```

Dieses Programm wird vor dem Aufruf von CP/M unter Basic gestartet, wobei der Schneider Drucker natürlich schon eingeschaltet sein muß. Der Drucker weiß jetzt auch noch unter Wordstar, daß er Texte im Schönschrift-Modus und mit 1 1/2 Zeilen Abstand ausgeben soll. Wer einen MBASIC-Compiler für CP/M besitzt, kann das Ganze auch noch einfacher haben:

```
10 LPRINT CHR$(27);"I";CHR$(3);
20 LPRINT CHR$(27);"3";CHR$(51);
```

Nach dem Compilieren und Linken steht das Programm als COM-File auf der Diskette und kann direkt von Wordstar aus mit dem (R)un-Befehl aufgerufen werden.

Cursor-Steuerung

Ein anderes Problem ist die Cursor-Steuerung. Besitzer einer Wordstar 3.0 Version, bei denen lediglich die Bildschirmsteuerung angepaßt wurde, können zunächst die Cursortasten nicht wie gewohnt benutzen. Dieses Problem läßt sich aber relativ einfach, mit Hilfe der Dienstprogramme SETUP unter AMSDOS bzw. INSTALL unter VDOS von VORTEX lösen. Dabei müssen nur noch die Tastennummern 0,1,2 und 8 mit den Controlsequenzen E,D,X und S belegt werden. Je nach Wunsch kann auch noch der Funktionstastenblock belegt werden.

Wordstar-Aufsteiger

Viele Aufsteiger, die vor Wordstar mit anderen Textverarbeitungsprogrammen gearbeitet haben, möchten ihre Texte oder Berichte wahrscheinlich nicht noch einmal neu eintippen. Wenn diese Texte als ASCII-Dateien vorliegen, wie sie z.B. unter TASWORD erstellt werden, gibt es hier keine Probleme. Deutsche Sonderzeichen oder bestimmte Druckeransteuerungszeichen lassen sich nach dem Hereinladen bequem und einfach mit dem Control QA-Menue bearbeiten und entsprechend umwandeln. Bequemer geht es kaum noch.

Der leidige Bildschirmaufbau

Der Bildschirmaufbau ist unter Wordstar leider sehr langsam und kann selbst durch die schnellere Bildschirmausgabe unter VDOS 2.0 von VORTEX durch den neuen CP/M-Befehl FAST nur geringfügig beschleunigt werden. Man muß jedoch nicht auf den vollständigen Bildschirmaufbau warten, sondern sollte immer munter drauflos tippen und den Tastaturpuffer nutzen.

VORTEX hat bereits einen Wordstar-Patch mit einer wesentlich schnelleren Bildschirmausgabe angekündigt.

Speicherplatz unter Wordstar

Wie bereits schon erwähnt, läuft Wordstar 3.0 auch ohne die Speichererweiterung von VORTEX auf den CPC's.

Diese Tatsache ist unter anderem dem Umstand zu verdanken, daß vor einigen Jahren, als Wordstar für CP/M-Rechner auf den Markt kam, schon aus Kostengründen mit dem Speicherplatz gespart werden mußte und viele Rechner mit einem kleineren RAM als 64KB ausgestattet waren. Wordstar mußte also an die Gegebenheiten angepaßt werden. Das komplette Wordstar umfaßt allerdings immer noch fast 100KB, wobei jedoch durch eine Overlay-Technik immer nur der gerade benötigte Teil in den RAM-Speicher hereingeladen wird. Ebenso verhält es sich mit den zu bearbeitenden Texten und Programmen, wodurch die Länge dieser Texte im Prinzip nur von der Speicherkapazität der Diskette abhängt. Dabei wird ständig Text von der Originaldatei kopiert und in den, je nach Speichergröße, verbleibenden RAM-Bereich geladen. Wird der Cursor um mehrere Zeilen nach unten bewegt, muß wieder ein neuer Teil nachgeladen werden. Dabei wird der gerade im Speicher befindliche Teil jedoch nicht einfach gelöscht, sondern auf einer neuen Datei zwischengespeichert. Auch beim Rückwärtsgehen mit dem Cursor über weitere Strecken, wird wieder eine neue Datei angelegt. Das ist übrigens auch der Grund dafür, warum es beim Arbeiten unter Wordstar ständig zu Diskettenzugriffen kommt. Beim 3"-Laufwerk stößt man mit dieser zwar sehr sicheren aber verschwenderischen Methode ziemlich schnell an seine Grenzen, so daß man bei einer Textlänge von 15KB so langsam aufhören muß. Der restliche Platz wird für die Sicherheitskopien benötigt und vor allen Dingen für Wordstar selber, das sich ja immer auf der Arbeitsdiskette befinden muß. Einen Ausweg bietet das Aufteilen in viele kleinere Textbausteine oder das Arbeiten mit einem Zweitlaufwerk. Viel eleganter arbeitet man allerdings mit den Vortex 5,25"- bzw. neuen 3,5"-Laufwerken, die ja beide schon als Einzellaufwerke eine Speicherkapazität von je 708KB besitzen, also das Vierfache einer 3"-Seite.

Um nun auch noch die, durch den sich



ständig wiederholenden Diskettenzugriff erzeugten, störenden Laufwerksgeräusche und die sich dadurch zwangsläufig ergebenden Verzögerungszeiten zu vermeiden, gibt es eine Ideallösung:

Die RAM-Disk als Zweitlaufwerk

Die von Vortex angebotene Speicheraufrüstung SP 64 erfreut den Wordstar-Benutzer zunächst einmal durch einen Drucker-SPOOLER und die Tatsache, daß mehr Text auf einmal in den RAM-Speicher hereingeladen werden kann.

Auch Blockverschiebungen müssen jetzt nicht mehr über die Diskette abgewickelt werden. Das Simultan-Drucken ist ebenfalls möglich.

Noch viel interessanter wird das Arbeiten allerdings ab der SP 128, oder noch besser SP 256, da man bei beiden in den Genuß einer RAM-Disk kommt.

Da die Logik der Karte die Ausbaustufe selbsttätig erkennt, kann jeder Besitzer einer SP 64 seine Karte einfach mit Hilfe von Aufrüstungssätzen bis auf 512KB, davon 448KB RAM-Disk, aufrüsten.

Wie ein geölter Blitz wird Wordstar jetzt geladen. Dabei stört kein einziges Laufwerksgeräusch den Benutzer, und selbst größere Texte werden in Sekundenschnelle hereingeladen. Auch das Laden der Overlays geschieht in wenigen Augenblicken. Das einzige Hemmnis ist, nach wie vor, der ziemlich langsame Bildschirmaufbau. Dafür kann der Benutzer Wordstar während dem Ausdrucken von Texten verlassen und ein völlig anderes Programm aufrufen, oder nach Wordstar zurückgehen und einen neuen Text bearbeiten. Der SPOOLER macht's. Man kann übrigens auch die Zeit während des Ausdruckens dafür nutzen, um alle Texte auf die Diskette zurück zu schreiben und damit dauerhaft zu sichern. Dafür muß man Wordstar nicht einmal verlassen, sondern bedient sich der Menuegesteuerten o-Option.

Rechtschreibungsüberprüfung

Sicherlich ist den meisten Benutzern im Hauptmenue von Wordstar schon die Option (S)PELLSTAR aufgefallen. SPELLSTAR ist ein Rechtschreibungs-Prüfprogramm und wird getrennt von Wordstar verkauft. Man sollte jedoch vom Erwerb dieses Programmes absehen, da es nur amerikanische Wörter kennt, und daher nicht ein einziges deutsches Wort versteht. Spellstar kann zwar auch die deutsche Rechtschreibung erlernen, jedoch ist das ein sehr mühseliges Unterfangen. Entweder verzichtet man ganz auf diese Möglichkeit oder kauft sich ein deutsches Rechtschreibungs-Prüfprogramm, das dann ebenfalls aus dem Wordstar-Hauptmenue mit (R)un aufgerufen werden kann.
Zusammenfassend kann man sagen, daß man sich den Umgang mit Wordstar, je nach Lust und Geldbeutel, mehr oder weniger erleichtern kann. Wordstar bietet viele professionelle Möglichkeiten und muß daher als ein ausgereiftes Textverarbeitungsprogramm bezeichnet werden. Alle Möglichkeiten wird man sicherlich erst nach einer längeren Einarbeitungsphase anwenden können.

Dipl.-Ing. H. Scheruhn

WordStar anpassen

Sie besitzen eine WordStar-Version von Markt & Technik oder die eines anderen Anbieters und überlegen sich, ob die Anpassung für Ihren WordStar wohl so gut ist, wie Sie es sich vorstellen.

Vielleicht zweifeln Sie, ob alle Möglichkeiten ausgeschöpft sind, da Sie bei einem Bekannten eine Version gesehen haben, die offensichtlich besser funktioniert bzw. mehr Funktionen des Druckers ausnutzt, als es Ihr Programm zuläßt. Andererseits haben Sie schon gehört, daß man WordStar »patchen« kann. Wie, wo, was, zu welchem Zweck, mit welchem Ergebnis?

Lesen Sie diesen Artikel durch, vollziehen Sie die gemachten Vorschläge, spielen Sie damit herum, passen Sie Sequenzen Ihren Bedürfnissen (Drukker, Ausführungszeiten, Steuerzeichenkomfort etc.) an.

Sie werden sehen, daß es über die Möglichkeiten des Installationsprogramms hinaus noch eine ganze Reihe von »Kleinigkeiten« gibt, um die Sie Ihr »persönliches« WordStar-Programm verfeinern können.

Bevor wir loslegen, will ich Ihnen zeigen, welche Möglichkeiten sich zum Beispiel für Ihre Textverarbeitung unter WordStar ergeben:
Beispiele für die Möglichkeiten, die Sie nach dem »patchen« bei einem EPSON-Drucker (oder E-kompatiblen Drucker) haben:

```
Beispiele für die Möglichkeiten, die
Sie nach dem 'patchen' bei einem
EPSON-Drucker (oder E-kompatiblen
Drucker) haben:

^PA = ESCANFANG / ^PN = ESCENDE

^PAM / ^PNP
-----------------------> Elite
^PA4 / ^PN5
-----------------------> Italic
^PA4^PNM / ^PA5^PNP
-----------------------> Italic+Elite
^PAp1 / ^PNp0
-----------------------> Proportional
^PW^PAp1 / ^PNp0^PQ
-----------------------> Prop.weit
^PE / ^PR
-----------------------> Engschrift
^PW / ^PQ
-----------------------> Weit
^PW^PE / ^PR^PQ
-----------------------> Sperrschrift
^PD^PW / ^PQ^PD
-----------------------> weit+dp.
^PW^PA4 / ^PN5^PQ
-----------------------> It. weit
^PT / ^PT
-----------------------> Tiefgestellt
^PY / ^PY
-----------------------> Hochgestellt
^PB / ^PB
-----------------------> Fett (bold)
^PD / ^PD
-----------------------> Doppelt
^PB^PA4 / ^PN5^PB
-----------------------> Italic fett
^PT^PA4 / ^PN5^PT
-----------------------> Italic tief
^PY^PA4 / ^PN5^PY
-----------------------> Italic hoch
^PT^PAM / ^PNP^PT
-----------------------> Elite tief
^PY^PAM / ^PNP^PY
-----------------------> Elite hoch
^PB^PAM / ^PNP^PB
-----------------------> Elite fett
^PW^PAM / ^PNP^PQ
-----------------------> Elite weit
^PT^PE / ^PR^PT
-----------------------> eng tiefgestellt
^PY^PE / ^PR^PY
-----------------------> eng hochgestellt
^PAn / ^PNP
-----------------------> Near Letter
^PA-1 / ^PN-0
-----------------------> Unterstrichen

--------------------------------------
kein Elite+Eng,   kein Elite+fett,
kein Elite+Sperr, kein Elite+Prop.
--------------------------------------
```

Welche Kenntnisse benötigen wir, um die nachfolgend beschriebenen WordStar-Adressen des 'ws.com'-Files zu verändern?

Nun, lediglich die Bedienung der Utilities DDT und SAVE unter CP/M 2.2! Mit 'ddt ws.com' laden wir unser WordStar-Programm und den »Debugger« in den Speicher und erhalten etwa die folgende Meldung:

DDT VERS 2.2
NEXT PC
4500 0100
-

Wir müssen uns den Wert unter 'NEXT' (hier 4500) merken, damit wir nach Abschluß unserer Arbeit mit DDT die richtige Speicherbereichslänge mit SAVE auf die Diskette zurückschreiben können.



4500 = Endadresse von 'ws.com'
0100 = Anfangsadresse von 'ws.com'
'-' = DDT-Promptzeichen

Mit der Ausgabe des Promptzeichens zeigt DDT an, daß nun Debugger-Kommandos erwartet werden. Uns interessieren aus der gesamten Kommando-Menge lediglich:

-d** (Dump ab Adresse ****)**
-s** (Verändern Inhalt Adresse)**
-g0 (verlassen DDT, Warmstart).

Mit 'd****' geben wir den Befehl DUMP ab Adresse **** (hier geben Sie natürlich eine »echte« Adresse ein, nicht etwa die Sternchen). DDT gibt Ihnen die nächsten 128 Speicherinhalte aus.

Mit 's****' wird die Adresse angewählt, deren Inhalt verändert werden soll. Dahinter erwartet DDT entweder einen zweistelligen Hexa-Wert oder einen Punkt ('.'), wenn der alte Wert erhalten bleiben soll.

Mit 'g0' verlassen wir DDT, indem wir das Programm mit der Warmstartadresse von CP/M fortsetzen. Danach saven wir den Speicherbereich, den 'ws.com' einnimmt, mit dem CP/M-Kommando SAVE.

SAVE 68 ws.com

Ihre Anzahl »Pages« (hier 68) ergibt sich aus dem Dezimalwert der ersten beiden NEXT-Wertstellen (hier 45H).

Die Anzahl muß um 1 vermindert werden, wenn die beiden letzten NEXT-Wertstellen gleich 00H sind (hier: 45H entsprechen '69' dezimal, vermindert um 1, also = 68!).

```
Name   Hex. Anz.   Inhalt     (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
DELCUS 02AE 1      00
   Zeitverz. in ms nach der Cursor-
   Adressierung.
DELMIS 02AF 1      00
   Zeitverz. in ms nach einer belie-
   bigen Zeichenausgabe.
CRBLIV 02B5 1      FF
   Cursor blinkt, wenn er in einer
   Invers-Darstellung steht (00 =
   'blinken' unterdrücken, FF =
   blinken).
DEL1   02CF 1      05
   Blinkdauer ('Cursor an'-Zeit)
   (wenn 02B5=FF).
DEL2   02D0 1      05
   Blinkdauer ('Cursor aus'-Zt.)
   (wenn 02B5=FF).
DEL3   02D1 1      00
   Zeit bis zum Erscheinen des Help-
   Menues.
DEL4   02D2 1      00
   Dauer der 'Copyright'-/'New-
   File'-Meldung.
DEL5   02D3 1      00
   Zeit bis zum Bildschirm-Aufbau
   nach horizontalem Scroll.
DEFDSK 02DC 1      01
   Laufwerk, auf dem WSMSGS.OVL und
   WSOVLY1.OVL zuerst gesucht werden
   sollen (01=A,02=B,03=C).
ITHELP 0360 1      01
   Hilfsstufe, mit der gestartet
   werden soll (00-03).
NITHLF 0361 1      FF
   Hinweis auf Help-Funktion unter-
   drücken bei WS-Start? (nein/ja)
INITOG 0362 1      FF
   Insert-Modus an bei WS-Start?
   (00=nein, FF=ja)
ITDSDR 0363 1      FF
   Inhaltverzeichnis anzeigen bei
   WS-Start?      (00=nein, FF=ja)
```

Bei den nächsten Adressen handelt es sich um Standardeinstellungen. Dabei sind auch solche für Punktbefehle, deren Eintragungen aus vier Bytes bestehen:

- Anzahl Zeilen (1 Byte)
- Anzahl Zeilen*Zeilenabstand (2 Bytes)
- Zeilenabstand 1/48 Zoll (1 Byte)

```
Name   Hex. Anz.   Inhalt     (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
INITPF 0366 1      08
   Zeilenabst. 1/48 Zoll      (.LH)
       0367 4      48 40 02 08
   Blattlänge                 (.PL)
       036B 4      03 18 00 08
   Länge des oberen Randes    (.MT)
       036F 4      02 10 00 08
   Kopfzeilenabst.v.o.Rand    (.HM)
       0373 4      08 40 00 08
   Länge des unt. Randes      (.MB)
       0377 4      02 10 00 08
   Fußzeilenabst.v.u.Rand     (.FM)
       037B 1      00
   Alternate-Zeichenbreite an?
   (00=nein, FF=ja)
       037C 1      0C
   Normal-Zch.breite 1/120"   (.CW)
       037D 1      0A
   Alternativ-Zch.br. 1/120"  (.CW)
       037E 1      08
   Breite des li. Randes      (.PO)
INITLM 037F 1      00
   Pos. der linken Randbegr.  (^OL)
INITRM 0380 1      42
   Pos. der rechten Randbegr. (^OR)
INITSR 0381 1      03
   Pos. hoch/tiefgest. Zeichen (.SR)
```

Wie bereits aus vorhergehenden Eintragungen zu ersehen ist, werden Fragen mit den Werten '00' für 'nein' und 'FF' für 'ja' beantwortet.

```
Name   Hex. Anz.   Inhalt     (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
INITWF 0385 1      FF
   Wortumbruch an?            (^OW)
       0386 1      FF
   Blocksatz an?              (^OJ)
       0387 1      FF
   Variabler Tabulator an?    (^OV)
       0388 1      00
   Phantom-Trennstrich an?    (^OE)
       0389 1      FF
   Trennhilfe an?             (^OH)
       038A 1      FF
   Anzeige Steuerzch. an?     (^OD)
       038B 1      FF
   Lineal-Anzeige an?         (^OT)
       038C 1      FF
   Dyn. Seitenanzeige an?     ---
       038D 1      FF
   Seiten-Anzeige?            (^OP)
       038F 1      00
   Standard-Blockverarbeitung (^KN)
   (00=Blockverarb.,  FF=Spaltenv.)
NONDOC 0392 1      00
   Programm-Modus an?         (N)
DECCHR 0393 1      2C
   Dezimalpunkt = ASCII-Code '2C'
DOTCHR 0395 1      2E
   Punkt = ASCII-Code '2E'
BLNCHR 0396 1      0F
   ASCII-Wert für CTRL O (^O)
DOTSON 0397 1      FF
   Punktkommandos wirksam? (nein/ja)
HZONE  039A 1      04
   Max. Anzahl von Leerzeichen, die
   bei ^B eingefügt werden sollen,
   bevor ein Trennvorschlag gemacht
   wird.
```

```
Name   Hex. Anz.   Inhalt     (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
PWVTAB 039B 2      9F 03
   Adresse der Tabelle für Umlaute
   oder andere Zeichen (VOWTAB), die
   bei der Silbentrennung wie Vokale
   behandelt werden sollen (ÄÖÜäöü).
PNCON  039D 2      A5 03
   Adresse der Tabelle für Vokale
   (Nichtkonsonanten) oder andere
   Zeichen (NONCON); wichtig für die
   Silbentrennung (AEIOUY).

VOWTAB 039F n      5B 5C 5D 7B 7C 7D

NONCON 03A5 n      41 45 49 4F 55 59

EOFCHR 03AD 1      2E
   '.' = Anzeige 'Ende des Textes'
BOFCHR 03AE 1      3A
   ':' = Anzeige 'Anfang des Textes'
CONCHR 03AF 1      2B
   '+' = Anzeige 'Text reicht über
         den rechten Rand'
OVPCHR 03B0 1      2D
   '-' = Anzeige 'die nächste Zeile
         überschreibt diese'
LFCHR  03B1 1      4A
   'J' = Anzeige 'Zeilenvorschub'
PAGCHR 03B2 1      50
   'P' = Anzeige 'Seitenumbruch'
SOFTCR 03B3 1      20
   ' ' = Anzeige 'Zeilenumbruch'
HARDCR 03B4 1      3C
   '>' = Anzeige 'Neuer Absatz'
FDTCHR 03B5 1      4D
   'M' = Anzeige 'Mail-Merge-KDO.'
SOFHYC 03B8 1      AD
   '-' = Anzeige 'Phant.-Bindestr.'
PAGFIL 03B9 1      2D
   '-' = Anzeige 'Seitenumbruch'
MARKS  03BA 1      42
   'B' = Anzeige 'Blockanfang'
       03BB 1      4B
   'K' = Anzeige 'Blockende'
       03BF 10     30 31 32 33 34
                   35 36 37 38 39
   '0123456789'=Anzeige der Block-
               Merker 0-9
```

Die folgenden Adressen beinhalten Daten für Druck-Standard-Einstellungen (auch hier gilt '00' für 'nein' und 'FF' für 'ja').

```
Name   Hex. Anz.   Inhalt     (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
Die folgenden Adressen beinhalten
Daten für Druck-Standard-Einstellun-
gen (auch hier gilt '00' für 'nein'
und 'FF' für 'ja').

PODBLK 03CA 1      00
   Ausgabe auf Diskette?
       03CB 1      00
   Form-Feed?
       03CC 1      00
   Seitenformatierung unterdrücken?
```




```
       03CD 1        00
   Einzelblatteinzug  (Pause  bei
   Blatt-Ende)?
NOUFF  03D1 1        00
   Unterdrücken 'Seitenvorschübe' im
   Druck-Dialog?
ITPOPN 03D3 1        00
   Seitennummer unterdrücken?  (.OP)
ITMIJ  03D4 1        FF
   Microjustification an?      (.UJ)
ITBIP  03D5 1        FF
   Bidirektional drucken?      (.BP)
RVELIM 03D8 1        2C
   ',' = Trennzeichen zwischen den
         Elementen in  Mail-Merge-
         Dateien
RVQUOT 03D9 1        22
   '"' = Anführungszch. bei Mail-M.
VARCH1 03DB 1        26
   '&' = Variablen-Start bei Mail-M.
VARCH2 03DC 1        26
   '&' = Variablen-Ende  bei Mail-M.
VARNBC 03DD 1        2A
   '*' = Zeichen hinter dem ersten
         '&' zum Unterdrücken von
         Leerzeilen, wenn Variable=0
         bei Mail-Merge
VAROPC 03DE 1        2F
   '/' = Trennzeichen zwischen Var.
         und Option (bei Mail-Merge)
       03DF 1        4F
   'O' = Option vor dem 2. '&' zum
         Unterdrücken von Leerzeilen
         bei entfallenden Variablen
         (bei Mail-Merge)
FNWSCM 03E6 12       00 57 53 20 20 20
                     20 20 20 43 4F 4D
   Name des WS-Files 'WS.COM'
DSKTNA 03F2 12       00 57 53 4D 53 47
                     53 20 20 4F 56 52
   Name des WS-Files 'WSMSGS.OVR'
FNOVLY 03FE 12       00 57 53 4F 56 4C
                     59 31 20 4F 56 52
   Name des WS-Files 'WSOVLY1.OVR'
FNMRGP 040A 12       00 4D 41 49 4C 4D
                     52 47 45 4F 56 52
   Name des WS-Files 'MAILMRGE.OVR'
```

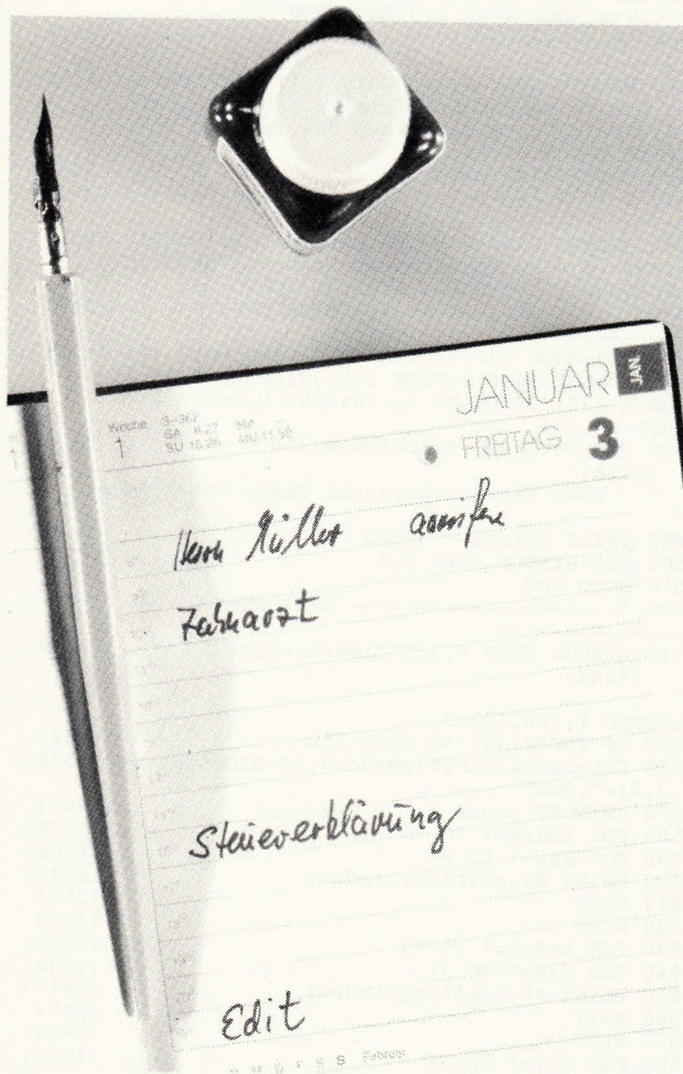
Im folgenden sind druckspezifische Daten angegeben. Belegt eine Position mehr als ein Byte, so gibt das erste Byte immer die Anzahl der Bytes an, aus der eine Sequenz besteht.

```
Name   Hex. Anz.   Inhalt      (Hexa.)
Param. Adr. Bytes  ## ## ## ## ## ##
-------------------------------------
Im Folgenden sind Druck-spezifische
Daten angegeben. Belegt eine Posi-
tion mehr als ein Byte, so gibt das
erste Byte immer die Anzahl der
Bytes an, aus der eine Sequenz be-
steht.

POSMTH 0690 1        FF
   Art des Unterstreichens:
   00=BS und Unterstreichungszeichen
   01=Typenraddrucker
   FF=CR mit Unterstreichungs-Zeile
BLDSTR 0691 1        04
   Anzahl 'überdrucken' bei Fett-
   Druck                        (^PB)
DBLSTR 0692 1        02
   Anzahl 'überdrucken' bei Doppel-
   Druck                        (^PD)
PSCRLF 0696 11       02 0D 0A
   ASCII-Code für CR und LF
PSCR   06A1 7        01 0D
   ASCII-Code für CR

PALT   06B5 5        01 1B
   ESC-Seq. 'Sonderschrift ein'(^PA)
PSTD   06BA 5        01 1B
   ESC-Seq. 'Sonderschrift aus'(^PN)
ROLUP  06BF 5        03 1B 53 01
   ESC-Folge: ESC S 1
   'Tiefstellen ein'            (^PT)
ROLDOW 06C4 5        04 1B 54 1B 48
   ESC-Folge: ESC T + ESC H
   'Tiefstellen aus'            (^PT)
USR1   06C9 5        03 1B 57 00
   ESC-Folge: ESC W 0
   'Weit aus'                   (^PQ)
USR2   06CE 5        03 1B 57 01
   ESC-Folge: ESC W 1
   'Weit ein'                   (^PW)
USR3   06D3 5        01 0F
   ESC-Folge: SI
   'Eng ein'                    (^PE)
USR4   06D8 5        03 12 1B 50
   ESC-Folge: DC + ESC P
   'Eng aus+Normal'             (^PR)
RIBBON 06DD 5        03 1B 53 00
   ESC-Folge: ESC S 0
   'Hochstellen ein'            (^PY)
RIBOFF 06E2 5        04 1B 54 1B 48
   ESC-Folge: ESC T + ESC H
   'Hochstellen aus'            (^PY)
PSINIT 06E7 17       07 1B 40 1B 52 01
                     1B 4F
   ESC-Fo.: ESC Kl.af.+ESC R 1+ESC O
   'Drucker INIT'
PSFINI 06F8 17       06 1B 57 00 12 1B
                     40
   ESC-Folge: ESC W 0+DC2+ESC Kl.af.
   'Drucker END'
```

H. Kleiner



Terminkalender

für 464-664-6128



Dieses Basicprogramm ist den Vergeßlichen unter uns gewidmet, die entweder zu spät oder gar nicht zu einem schon lange geplanten Termin erscheinen, weil er mangels Planung in Vergessenheit geriet. Mit dem TERMINKALENDER wird das Speichern, Ändern und Löschen von Terminen zur Wohltat, da dieses Programm über einige komfortable »Nebensächlichkeiten« verfügt.

Nach dem Laden des Hauptprogrammes von Diskette, wird der Benutzer nach dem Vornamen gefragt. Dieser Name wird dann beim Abspeichern eines Monats-Terminkalenders Bestandteil des Dateinamens, so daß auf einer Diskette die Terminkalender mehrerer Personen gespeichert werden können.

Anschließend wird das aktuelle Datum in vereinfachter Form eingegeben und nach dem aktuellen Wochentag gefragt. Aus diesen beiden Daten wird der gültige Monatskalender erstellt, der dann im Hauptmenue jederzeit abrufbar ist. Zum Schluß der Eingangsabfrage muß noch die Uhrzeit eingegeben werden; während der Arbeit wird dann die Uhrzeit in der Kopfzeile angezeigt.

Das Hauptmenue verfügt über alle zum Bearbeiten der Termine notwendigen Funktionen, so können neue Termine unter Angabe von Uhrzeit und Notiz eingetragen werden, bestehende können geändert, zeitlich verlegt oder ganz gelöscht werden. Nach dem Editieren der Termine kann der Kalender, wie oben beschrieben, unter Verwendung des Vornamens abgespeichert werden. Zum Laden wird die Angabe des gewünschten Tages benötigt; geladen wird aber der Terminplan des ganzen Monats. Mit den '+/−'-Tasten kann ein Kalender tageweise durchgeblättert werden, was die Suche nach einem bestimmten Termin sehr erleichtert. Schließlich kann das Inhaltsverzeichnis der Diskette per Tastendruck eingelesen werden.

Die eingegebenen Termine werden vom TERMINKALENDER automatisch sortiert, so daß man sich bei der Eingabe der Uhrzeit nicht an die zeitliche Reihenfolge der Termine halten muß. Es sei auch noch erwähnt, daß das Programm bei der Erstellung der Monatskalender automatisch die Schaltjahre berücksichtigt.

(V. Soergel/ME)

```
10 '************************************** [6030]
****************************************
*******          Terminkalender   Version 2.
1      22.10.1985              *******
20 '*******  (C) Volker Soergel          [6727]
    4050 Moenchengladbach 2       *******
****************************************
************************************
30 OPENOUT"D":MEMORY HIMEM-1:CLOSEOUT    [2548]
40 termzahl=15:DIM zeit(31,termzahl):DIM b [8346]
em$(31,termzahl):DIM last(31):DIM woch$(7)
:DIM daymax(12)
50 woch$(1)="Montag":woch$(2)="Dienstag":w [13519]
och$(3)="Mittwoch":woch$(4)="Donnerstag":w
och$(5)="Freitag":woch$(6)="Samstag":woch$
(7)="Sonntag"
60 daymax(1)=31:daymax(2)=28:daymax(3)=31: [13443]
daymax(4)=30:daymax(5)=31:daymax(6)=30:day
max(7)=31:daymax(8)=31:daymax(9)=30:daymax
(10)=31:daymax(11)=30:daymax(12)=31
70 MODE 2:INK 0,0:INK 1,26               [2703]
80 WINDOW #1,1,80,1,3:PAPER#1,1:PEN#1,0  [2241]
90 WINDOW #2,1,80,4,25:PAPER#2,0:PEN#2,1 [2652]
100 WINDOW #3,1,80,24,25:PAPER#3,1:PEN#3,0 [3302]
110 WINDOW #4,1,18,7,22:PAPER#4,1:PEN#4,0 [2397]
120 WINDOW #5,22,80,5,22:PAPER#5,0:PEN#5,1 [2284]
130 WINDOW #6,3,18,5,5:PAPER#6,0:PEN#6,1 [2400]
140 WINDOW SWAP 0,1                      [1031]
150 CLS#0:LOCATE 1,2:PRINT"   *** T e r m i [6391]
 n k a l e n d e r ***
  ";CHR$(164);" Volker Soergel  "
160 WINDOW SWAP 0,2                      [1026]
170 LOCATE 1,10:INPUT"Wie lautet Ihr Vorna [3320]
me ";person$
180 LOCATE 1,12:INPUT"Bitte geben Sie das [12660]
heutige Datum ein  (z.B. 300885=30.08.85)
";datum$:day=VAL(LEFT$(datum$,2)):month=VA
L(MID$(datum$,3,2)):IF (VAL(RIGHT$(datum$,
2)) MOD 4)=0 THEN daymax(2)=29
190 FOR index=1 TO 31:last(index)=1:NEXT [2831]
200 LOCATE 1,14:INPUT"Welcher Wochentag "; [9138]
hilf$:index=1:WHILE UPPER$(hilf$)<>UPPER$(
woch$(index)) AND index<7:index=index+1:WE
ND:wotag=index
210 LOCATE 1,16:INPUT"Wie spaet ist es (z. [5866]
B. 12:10 Uhr=1210) ";timer$:std=VAL(LEFT$(
timer$,2)):minut=VAL(RIGHT$(timer$,2))
220 EVERY 3000 GOSUB 1240                [1446]
230 CLS                                  [91]
240 '.......................... TERMINW [5384]
INDOW AUFBAUEN ......................
250 CLS#6:LOCATE #6,2,1:PRINT #6,std:LOCAT [10003]
E #6,5,1:PRINT#6,":":LOCATE #6,6,1:PRINT#6
,USING"##";minut;:PRINT#6," Uhr":IF minut
<10 THEN LOCATE #6,6,1:PRINT#6,"0"
260 CLS#3:WINDOW SWAP 0,4:CLS:PRINT:PRINT: [15297]
onatskalender":PRINT:PRINT"
endern":
oeschen":PRINT:PRINT"
atalog":PRINT
 einen Tag"              [1792]
280 WINDOW SWAP 0,5                      [1043]
290 CLS:MOVE 160,40:DRAWR 0,300:DRAWR 479, [6747]
0:DRAWR 0,-300:DRAWR -479,0:MOVE 160,315:D
RAWR 479,0:MOVE 235,40:DRAWR 0,300
300 MOVE 7,340:DRAWR 115,0:DRAWR 0,-25:DRA [3970]
WR -115,0:DRAWR 0,25
310 LOCATE 1,1:PRINT"Uhrzeit":LOCATE 11,1: [10840]
PRINT"Bemerkung         Datum: ";woch$(wota
g);",";STR$(day);".";MID$(datum$,3,2);".19
";RIGHT$(datum$,2)
320 FOR index=1 TO last(day)-1:LOCATE 2,in [9109]
dex+2:PRINT USING"##.##";zeit(day,index):L
OCATE 11,index+2:PRINT bem$(day,index):NEX
T
330 frag$=UPPER$(INKEY$):IF frag$="" THEN [2566]
330
340 IF frag$="E" THEN 360 ELSE IF frag$="A [13795]
" THEN 410 ELSE IF frag$="L" THEN 430 ELSE
 IF frag$="S" THEN 520 ELSE IF frag$="D" T
HEN 730 ELSE IF frag$="+" THEN 980 ELSE IF
 frag$="-" THEN 1010 ELSE IF frag$="M" THE
N 1290 ELSE IF frag$="C" THEN 450 ELSE 330
350 '-------------------- EINGABE -------- [3235]
------------
360 WINDOW SWAP 0,3:CLS                  [1833]
370 IF last(day)<=termzahl THEN LOCATE 4,1 [16312]
:INPUT"Uhrzeit des neuen Termins ";zeit(da
y,last(day)):LOCATE 4,2:PRINT"Bemerkung zu
m neuen Termin ? ";:LINE INPUT bem$(day,la
st(day)):IF LEN(bem$(day,last(day)))>48 TH
EN GOSUB 1120:GOTO 370
380 IF last(day)<=termzahl THEN CLS:GOSUB [5371]
1060:last(day)=last(day)+1:WINDOW SWAP 0,3
:GOTO 290
390 IF last(day)>termzahl THEN LOCATE 10,2 [8409]
"
:SOUND 1,100,100,5:FOR index=1 TO 1000:NEX
T:CLS:WINDOW SWAP 0,3:GOTO 290
400 '-------------------- AENDERN -------- [2619]
------------
410 WINDOW SWAP 0,3:LOCATE 4,1:INPUT"Uhrze [9396]
it des zu aendernden Termins ";hilf:LOCATE
 4,2:PRINT"Neue Bemerkung des Termins ? ";
:LINE INPUT hilf$:CLS
420 GOSUB 1140:GOTO 290                  [955]
430 '-------------------- LOESCHEN ------- [2176]
-------------
440 WINDOW SWAP 0,3:CLS:LOCATE 4,1:INPUT"U [8316]
hrzeit des zu loeschenden Termins ";hilf:C
LS:GOSUB 1170:WINDOW SWAP 0,3:GOTO 290
450 '-------------------- DIRECTORY ------ [3357]
--------------
460 CLS:MOVE 639,340:DRAWR 0,-300:MOVE 161 [6204]
,315:DRAWR 10,0,0:MOVE 235,42:DRAWR 0,10,0
:MOVE 235,339:DRAWR 0,-10:MOVE 0,0:DRAWR 0
,0,1
470 CAT:MOVE 639,50:DRAWR 0,250          [1579]
 [5838]
      Fuer Terminuebersicht Taste druecken
"
490 WHILE INKEY$="":WEND                 [1607]
500 CLS:WINDOW SWAP 0,3                  [484]
510 GOTO 290                             [411]
520 '-------------------- SPEICHERN ------ [1711]
--------------
530 WINDOW SWAP 0,3:CLS:merke=day        [2842]
 [4646]
"
:SOUND 1,100,100,5
550 IF INKEY(45)<>0 THEN 550             [582]
560 CLS:name$=LEFT$(person$,3)+MID$(datum$ [3503]
,3,2)+".TER"
570 OPENOUT name$                        [980]
580 FOR index=1 TO 15                    [1336]
590 FOR day=1 TO 31                      [1588]
600 PRINT #9,zeit(day,index)             [2726]
610 NEXT                                 [350]
620 NEXT                                 [350]
630 FOR index=1 TO 15                    [1336]
640 FOR day=1 TO 31                      [1588]
650 PRINT #9,bem$(day,index)             [1947]
660 NEXT                                 [350]
670 NEXT                                 [350]
680 FOR day=1 TO 31                      [1588]
```

```
690 PRINT #9,last(day)                          [2322]
700 NEXT                                        [350]
710 CLOSEOUT                                    [902]
  [9393]
":SOUND 1,
100,100,5:FOR index=1 TO 1000:NEXT:CLS:WIN
DOW SWAP 0,3:GOTO 290
730 '-------------------- LADEN ----------      [2677]
----------
740 WINDOW SWAP 0,3:CLS:LOCATE 4,1:INPUT"W      [18107]
elches Datum soll geladen werden ";datum$:
merke$=datum$:INPUT"   Was ist das fuer ei
n Wochentag ";hilf$:index=1:WHILE UPPER$(h
ilf$)<>UPPER$(woch$(index)) AND index<7:in
dex=index+1:WEND:wotag=index
750 name$=LEFT$(person$,3)+MID$(datum$,3,2      [3390]
)+".TER"
760 CLS:LOCATE 1,2:PRINT"                       [5939]
":SOUND
1,100,100,5
770 IF INKEY(45)<>0 THEN 770                    [810]
780 OPENIN name$                                [1156]
790 FOR index=1 TO 15                           [1336]
800 FOR day=1 TO 31                             [1588]
810 INPUT #9,hilf                               [891]
820 zeit(day,index)=hilf                        [1791]
830 NEXT                                        [350]
840 NEXT                                        [350]
850 FOR index=1 TO 15                           [1336]
860 FOR day=1 TO 31                             [1588]
870 INPUT #9,hilf$                              [994]
880 bem$(day,index)=hilf$                       [2001]
890 NEXT                                        [350]
900 NEXT                                        [350]
910 FOR day=1 TO 31                             [1588]
920 INPUT #9,hilf                               [891]
930 last(day)=hilf                              [2110]
940 NEXT                                        [350]
950 CLOSEIN                                     [752]
960 day=VAL(LEFT$(datum$,2)):month=VAL(MID      [14339]
$(datum$,3,2)):LOCATE 4,2:PRINT"
":SOUND 1,100,
100,5:FOR index=1 TO 1000:NEXT:CLS:WINDOW
SWAP 0,3:GOTO 290
970 '------------------- "DAY" VERAENDERN       [4077]
--------------------------
980 day=day+1:IF day>daymax(month) THEN da      [5307]
y=daymax(month):GOTO 290
990 wotag=wotag+1:IF wotag>7 THEN wotag=1       [3384]
1000 GOTO 290                                   [411]
1010 day=day-1:IF day<1 THEN day=1:GOTO 29      [1845]
0
1020 wotag=wotag-1:IF wotag<1 THEN wotag=7      [3332]
1030 GOTO 290                                   [411]
1040 END                                        [110]
1050 '...................... SORTIERROUT        [4091]
INE.........................
1060 FOR index2=1 TO 15:FOR index=1 TO las      [3505]
t(day)-1
1070 IF zeit(day,index)>zeit(day,index+1)       [14604]
THEN hilf=zeit(day,index):hilf$=bem$(day,i
ndex):zeit(day,index)=zeit(day,index+1):be
m$(day,index)=bem$(day,index+1):zeit(day,i
ndex+1)=hilf:bem$(day,index+1)=hilf$
1080 NEXT                                       [350]
1090 NEXT                                       [350]
1100 RETURN                                     [555]
1110 '......................."BEMERKUNG K       [4303]
UERZER".........................
     Die Bemer [8444]
":SOUND 1,100,10
0,5:FOR index=1 TO 1000:NEXT:CLS:RETURN
1130 '...................... AENDERN            [3919]
..........................
1140 FOR index=1 TO last(day)-1:IF zeit(da      [6566]
y,index)=hilf THEN bem$(day,index)=hilf$
1150 NEXT                                       [350]
1160 WINDOW SWAP 0,3:RETURN                     [2530]
1170 '...................... LOESCHEN           [3883]
..........................
1180 FOR index=1 TO last(day)-1:IF zeit(da      [7751]
y,index)=hilf THEN zeit(day,index)=0:bem$(
day,index)=""
1190 NEXT                                       [350]
1200 FOR index2=1 TO 15:FOR index=1 TO las      [14337]
t(day)-1:IF zeit(day,index)=0 THEN zeit(da
y,index)=zeit(day,index+1):bem$(day,index)
=bem$(day,index+1):zeit(day,index+1)=0:bem
$(day,index+1)=""
1210 NEXT                                       [350]
1220 NEXT                                       [350]
1230 last(day)=last(day)-1:RETURN               [3125]
1240 '...................... UHRZEIT .          [3138]
..........................
1250 minut=minut+1:IF minut>59 THEN minut=      [3874]
0:std=std+1
1260 IF std>23 THEN std=0                       [1518]
1270 LOCATE #6,2,1:PRINT#6,std:LOCATE #6,5      [9920]
,1:PRINT#6,":":LOCATE #6,6,1:PRINT#6,USING
"##";minut;:PRINT#6,"  Uhr":IF minut<10 TH
EN LOCATE #6,6,1:PRINT#6,"0":MOVE 122,340:
DRAWR 0,-25
1280 RETURN                                     [555]
1290 '...................... KALENDER ERR       [4542]
ECHNEN ........................
1300 CLS:LOCATE 20,1:PRINT"Kalender ";:PRI      [10632]
NT USING"##";month;:PRINT"/";RIGHT$(datum$
,2):LOCATE 1,4:FOR index=1 TO 7:PRINT"
     ";woch$(index):PRINT:NEXT
1310 MOVE 639,340:DRAWR 0,-300:MOVE 161,31      [6909]
5:DRAWR 10,0,0:MOVE 235,42:DRAWR 0,10,0:MO
VE 235,339:DRAWR 0,-10:MOVE 0,0:DRAWR 0,0,
1
1320 index=day:hilf=wotag                       [2412]
1330 WHILE index>1:index=index-1:hilf=hilf      [5220]
-1:IF hilf=0 THEN hilf=7
1340 WEND                                       [390]
1350 hilf$=woch$(hilf):spalte=27                [2945]
1360 FOR index=1 TO daymax(month)               [3291]
1370 zeile=3+2*hilf-1:LOCATE spalte,zeile:      [6296]
PRINT USING"##";index
1380 hilf=hilf+1:IF hilf>7 THEN hilf=1:spa      [4754]
lte=spalte+4
1390 NEXT                                       [350]
 [5838]
      Fuer Terminuebersicht Taste druecke
"
1410 WHILE INKEY$="":WEND                       [1607]
1420 CLS:WINDOW SWAP 0,3                        [484]
1430 GOTO 290                                   [411]
```

CP/M-schneidern auf de

Wollten Sie nicht schon längst einmal Ihr CP/M 2.2 aufpolieren, haben sich aber nicht getraut? Keine Angst, bei richtiger Anwendung von SETUP.COM kann das auch ein Anfänger. Meist gibt es Probleme, die englischen Anweisungen zu verstehen, aber hier wird Ihnen erklärt, wie Sie die gewünschten Farben für BORDER, PAPER und PEN unter CP/M einschalten und sich einen persönlichen Begrüßungsstring schaffen. Dann wird der CPC Sie z.B. mit "Hallo Norbert! - Ich erwarte Deine Befehle!" begrüßen.

Möchten Sie auch unter CP/M die Funktionstasten nach Ihren Wünschen belegen? Kein Problem, Sie können bis zu 32 Tasten in verschiedenen Ebenen (Normal, mit SHIFT-Taste, mit CTRL-Taste) belegen. Hier können Sie z.B. Ihren Absender eingeben, der dann auf Tastendruck erscheint.

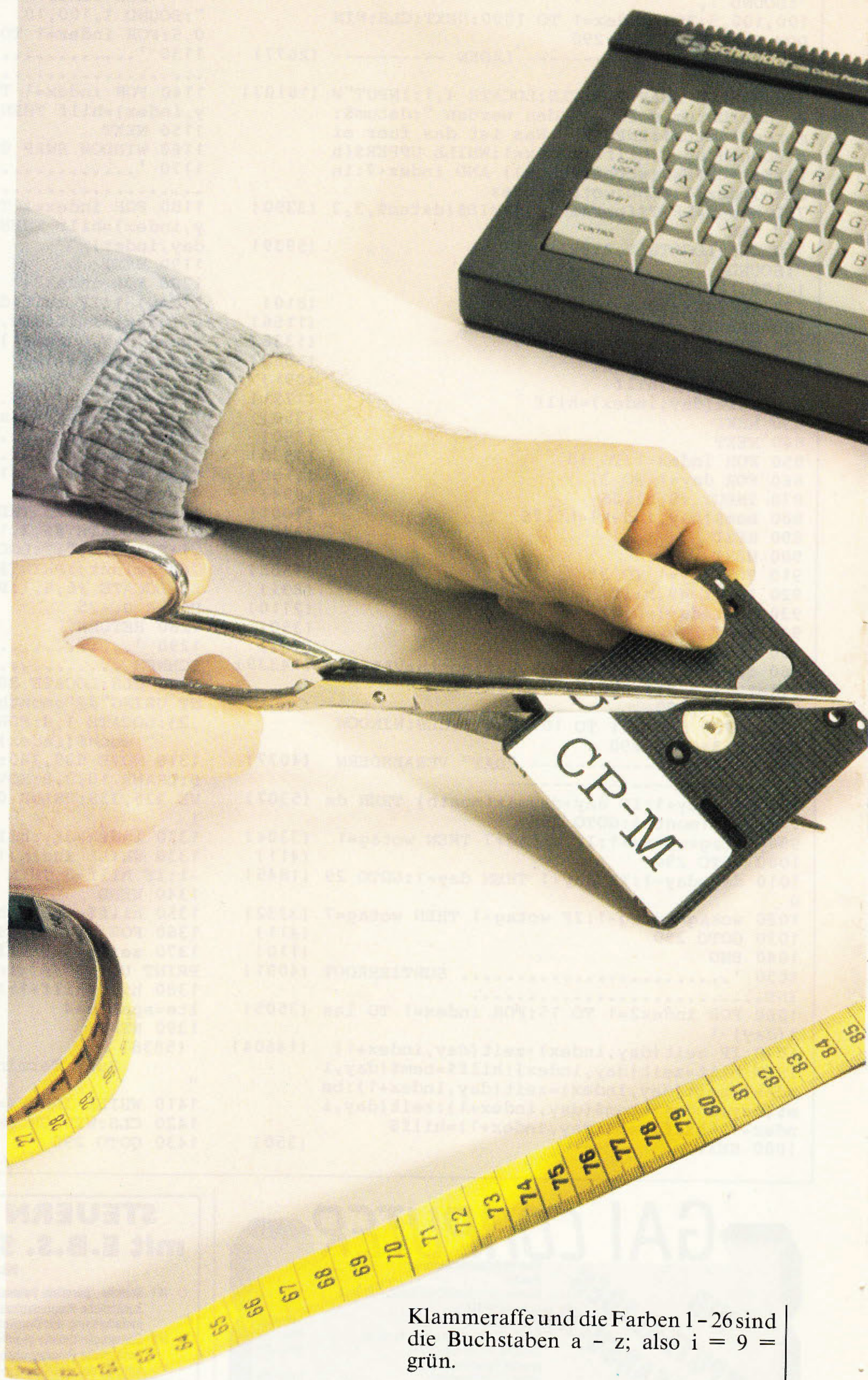
Wir wollen die Tastenbelegung so vornehmen, daß diese für WORDSTAR genutzt werden kann. Genauso kann die Tastatur aber an jedes andere CP/M-Programm angepaßt werden. Dieser erste Versuch sollte aber mit einer neuen, leeren Diskette vorgenommen werden. Formatieren Sie wie üblich mit dem Befehl FORMAT oder DISKIT2. Dann übertragen Sie mit FILECOPY SETUP.COM das SETUP-Programm auf die leere Diskette.

Nun rufen Sie den Befehl SETUP auf und Sie sehen die Meldung "Initial command buffer empty" und "Is this correct (Y/N):" Die Frage "Is this correct (Y/N):" erscheint bei jeder Meldung. Wenn Sie Y eingeben, geht es weiter zum nächsten Punkt. Wollen Sie etwas ändern, so geben Sie N ein.

Im "Initial command buffer" kann man Sofortbefehle ablegen, die mit dem Aufruf von CP/M ausgeführt werden. Geben Sie hier DIR^M ein, so wird immer zuerst das Directory angezeigt. Die sofortige Ausführung wird durch Hochpfeil + M (^M) erreicht, das entspricht der ENTER-Taste. Aber auch jedes Programm kann hier sofort gestartet werden. Meistens empfiehlt es sich, an dieser Stelle ein CP/M-Programm mit deutschen Sonderzeichen zu starten, z.B. DEUTSCH^M. Und auch das Hauptprogramm kann hier gestartet werden, bei WORDSTAR durch WS^M.

Drücken Sie ENTER, wenn alles eingegeben ist und bestätigen Sie die Frage: "Is this correct (Y/N):" nun mit Y. Mit der Meldung "sign-on string:" wird der CP/M-Begrüßungsstring dargestellt, wie er normal auf der Betriebsdiskette steht. Geben Sie wieder N ein und schauen sich den alten String genau an. Hochpfeil plus Schrägstrich nach rechts, plus Klammeraffe zeigen an, daß die folgenden zwei Buchstaben die PAPER-Farbe bestimmen.

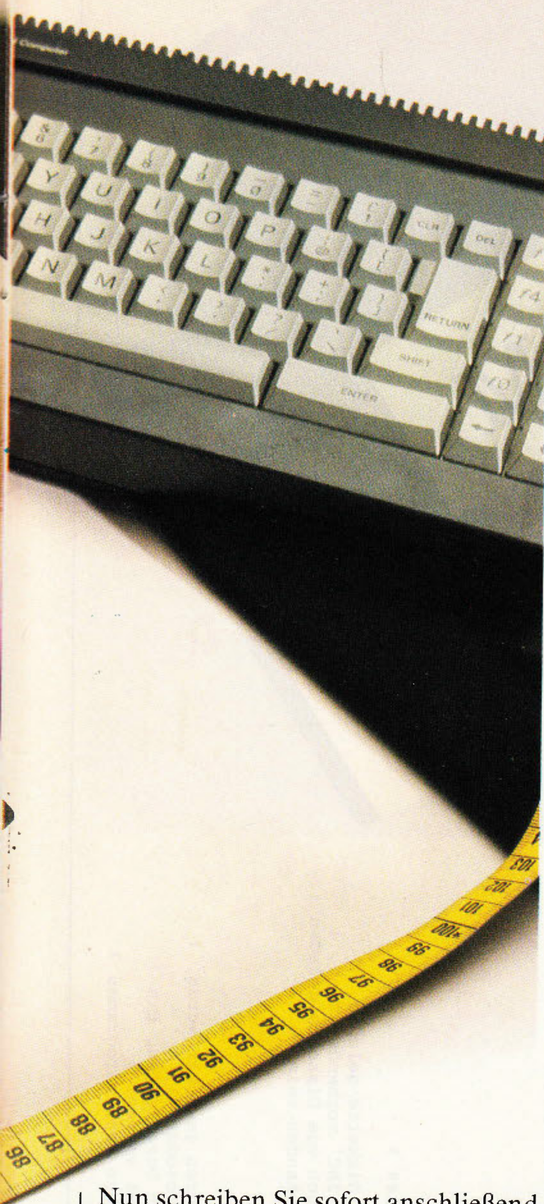
Wir wollen diese ändern und geben die ersten drei Zeichen so ein, wie sie schon



sind (Hochpfeil + Schrägstrich + Klammeraffe), dann zweimal den Buchstaben i. Diese beiden Buchstaben an der vierten und fünften Stelle wählen die Farbe aus, dabei ist die Farbe 0 der Klammeraffe und die Farben 1 - 26 sind die Buchstaben a - z; also i = 9 = grün.

Dann geben wir wieder Hochpfeil + Schrägstrich ein, gefolgt von dem Buchstaben a, der hier die PEN-Farbe anzeigt. Jetzt zweimal Klammeraffe eingeben für die PEN-Farbe 0. Nun geben wir Hochpfeil + eckige untere Klammer ein, das ist für die BORDER-Farbe. Hier können wir zweimal den Buchstaben c eingeben, für die BORDER-Farbe rot.

m Schneider



Nun schreiben Sie sofort anschließend Ihren eigenen Begrüßungstext, z.B. "CP/M 2.2 - Ich bin bereit! Du auch?", gefolgt von Hochpfeil + J und Hochpfeil + M (^J^M). Das setzt den Cursor in die nächste Zeile und führt den Befehl aus. Nach ENTER und Y übergehen wir den Abschnitt "Printer power-up string" und drücken noch einmal Y.

Hier erscheint die Meldung "Keyboard translations:" und wir geben wieder N ein, weil wir etwas eintragen wollen. Die Eingaben, die wir hier machen, entsprechen in etwa dem Befehl KEY DEF im Basic. Wir wollen hier eine wichtige Anpassung für WORDSTAR vornehmen. Die Pfeiltasten belegen wir so, daß das Editieren in Wordstar leichter fällt.

Jede Taste kann in drei Ebenen belegt werden. In der Normalebene können Sie mit den Cursortasten jeweils einen Buchstaben nach links und rechts bzw. eine Zeile nach oben und unten gehen. Mit der SHIFT-Taste springt der Cursor zum nächsten Wort links oder rechts bzw. zur nächsten oder vorherigen Seite. Wenn Sie die Control-Taste und Pfeil links bzw. rechts drücken, geht der Cursor an den Anfang bzw. an das Ende der Zeile. Mit CTRL plus Pfeil oben oder unten, wird der Text nach unten bzw. oben gescrollt.

Die Eingabe geschieht mit dem Buchstaben A, gefolgt von dem Tastenwert. Die Tastenwerte entnehmen Sie dem Anhang III, Seite 16 des Handbuches. Dann folgen die ASCII-Werte für die drei Ebenen. Eine Übersicht dieser Werte findet man im Handbuch, Anhang III, Seite 14. Besser ist der Überblick im Buch "CPC 464 inside out" von Huslik, auf den Seiten 390 - 393. Die Werte 1 bis 26 entsprechen der Eingabe von von CTRL+A bis CTRL+Z. Die Zeichen 64, 91 - 93 und 123 - 126 findet man auch in jedem guten Druckerhandbuch. Je nach Zeichensatz entsprechen sie dem Klammeraffen, den eckigen und geschwungenen Klammern, dem Schrägstrich nach rechts und dem geraden Strich bzw. den Umlauten ÄÖÜ äöü, dem ß und dem §.

Für die obere Pfeiltaste lautet die Eingabe:
A 0 5,18,141
Dabei entspricht 5 der CTRL+E-Taste (^E) und 18 der CTRL+R-Taste (^R). Der Wert 141 entspricht der Funktionstaste F13, diese wird erst im nächsten Abschnitt belegt. Alle 32 Funktionstasten (F0 - F31) haben die Werte 128 - 159 und diese Werte können auf beliebige Tasten gelegt werden. Um die Eingabe perfekt zu machen, geben Sie nun noch ENTER ein.

Die Belegung der Pfeiltasten erkennen Sie in der Tabelle an den Tastenwerte 0,1,2 und 8. Die COPY-Taste (9) wird als Ein-/Ausschalter für den Einfügemodus benutzt. Die DEL-Taste (79) wird mit ihrem normalen Wert zum Löschen nach links (127) belegt. CLR (16) löscht mit dem Wert 7 unter dem Cursor.

Die deutschen Umlaute können in CP/M nicht neu dargestellt werden, wie es in Basic mit dem Befehl SYMBOL geschieht. Diese Werte müssen mit einem File DEUTSCH.COM vorher eingeladen werden. Die Belegung der Tasten kann aber mit SETUP erfolgen. Wir legen diese Zeichen nun auf die Tasten 17, 19 und 22 (die beiden eckigen Klammern und der Schrägstrich nach rechts) in der Reihenfolge ä - ö - ü. Und zwar richtiger mit den Kleinbuchstaben in der Normal- und den Großbuchstaben in der SHIFT-Ebene. Auf der Taste 26 (Klammeraffe) liegt 'ß' und in der SHIFT-Ebene der '§'.

Machen Sie Ihre Eingabe nach der nun folgenden Tabelle, immer mit einem A beginnend und jede Zeile mit ENTER abschließen. Nach der letzten Eingabe noch einmal ENTER drücken. Dann sieht die Tabelle so aus:

Key code	Normal	Shift	Control
0	5	18	141
1	4	6	142
2	24	3	143
8	19	1	144
9	22	22	22
16	7	7	7
17	123	91	
19	124	92	
22	125	93	
26	126	64	
79	127	127	127

Wir schließen die Eingabe der gesamten Tabelle mit Y ab und gelangen zu "Keyboard translations:". Hier wollen wir nun die normalen Funktionstasten des Zahlenblocks belegen und die oben definierten zusätzlichen Funktionen auf den Pfeiltasten in der Control-Ebene. Zunächst N eingeben, dann wieder bei jeder Eingabe mit A beginnen und mit ENTER abschließen.
Zunächst legen wir für unsere Korrespondenz unseren vollständigen Absender und die Grußformeln auf den Zahlenblock. Wir beginnen mit dem Namen auf der 1 im Zahlenblock:

A 1,Schneider CPC International

Und so sieht die gesamte fertige Tabelle aus:

Expansions Token	Expansion String
1	Schneider CPC International
2	Fuldaer Straße 6
3	3440 Eschwege
4	Tel.: 05651/8702
7	Sehr geehrter Herr
8	Mit freundlichen Grüßen
13	^QW
14	^QD
15	^QZ
16	^QS

Nachdem wir entsprechend dieser Tabelle unsere eigene Adresse, die Begrüßungsformeln - nach eigenen Wünschen abzuändern (z.B. Hallo Fans!) - und die Belegung der Pfeiltasten eingegeben haben, drücken wir noch einmal ENTER und dann Y.

Alle anderen Abschnitte lassen wir ungeändert, indem wir immer Y eingeben. Bis wir dann zur Frage: "Do you want to update your system disc (Y/N):" kommen. Hier bestätigen wir mit Y und wenn wir gleich einen Probelauf wollen, dann auch die nächste Frage mit Y beantworten.

Nun haben Sie ein eigenes CP/M! Wenn Sie nun noch FORMAT.COM mit FILECOPY FORMAT.COM auf diese Diskette bringen, können Sie in Zukunft alle Disketten mit dem eigenen CP/M formatieren. Wo Sie die Belegung der Tasten nicht brauchen, können Sie diese weglassen oder abändern.

So sieht der komplette neue 'Sign-on string' aus:
^\@ii^\a@@^]ccCP/M 2.2 - Ich bin bereit! - Du auch?^J^M

H.-J. Behrendt

Fastcopy

Faster then PIP

Fastcopy ist ein schnelles Kopierprogramm für zwei Laufwerke. Die Anwendung dieses Programms als Einzel-File-Copy ist den Besitzern eines Zweitlaufwerkes vorbehalten, die beim Kopieren mit der CP/M-Masterdiskette bisher ein wenig stiefmütterlich behandelt wurden.
Weiterhin beinhaltet das Tool noch eine Disk-Utility, die das Editieren des Directorys wesentlich vereinfacht. Im Gegensatz zur bekannten 2-Drive-Kopiermethode mittels PIP, zeichnet sich Fastcopy durch anwenderfreundliche Menueführung und einer erheblichen Geschwindigkeitssteigerung von bis zu 40 % aus.



Anleitung:
Das Programm kann von CP/M aus unter dem Namen 'COPY' aufgerufen werden. Nach dem Aufruf meldet es sich mit dem Hauptmenue. Leicht kann man feststellen, daß zwei Unterprogramme aufgerufen werden können.

1. Das Kopierprogramm:
Dieses Kopierprogramm zeigt die Programme der Diskette im Laufwerk 'A' an und fragt, ob das angezeigte Programm kopiert werden soll. Diese Frage muß durch Drücken entweder der 'J'- oder der 'N'-Taste beantwortet werden. Nachdem alle Programme angezeigt wurden, beginnt das Programm zu kopieren. Die Kopierroutine endet mit der Frage, ob eine weitere Diskette bearbeitet werden soll. Wird diese Frage bejaht, so springt das Programm zum Hauptmenue zurück.

2. Das Löschprogramm:
Mit Hilfe dieses Programms ist es möglich, einzelne Pro-

```
{$U-} {Kompilerschalter zum Erreichen}
{$C-} {groesserer Geschwindigkeit}
PROGRAM Kopieren;

{ (c) Copyright 1986 by Arndt Grass }
{ Many thanks to my friends especially to Brocken }
{ Das Kopierprogramm arbeitet nur mit zwei Laufwerken unter CP/M }
{ Das Kopierprogramm muss als Com-File auf Diskette geschrieben werden, indem }
{ der Menupunkt Compileroption aufgerufen wird. In diesem Menu muss die }
{ 'C'-Taste betaetigt werden, was eine Umschaltung auf Com-File bewirkt. }
{ Nun kommt man durch druecken der 'q'-Taste wieder in das Hauptmenu zurueck. }
{ Wenn man nun die 'C'-Taste betaetigt wird das Program auf Disk kompiliert. }
{ Es kann spaeter durch eingeben des Namens unter CP/M aufgerufen werden. }
{ Viel Spass !!!!! }

CONST
 Buffergroesse   =200;   {Groesse des Zwischenspeichers, unter CP/M 3.0 darf die}
 Sectorgroesse   =128;   {Zahl fuer die Buffergroessse groesser sein}

TYPE
 Str15                = STRING [15];
 Str4                 = STRING [4];
 Schleifenvariablen = integer;

VAR
 Quelle,
 Ziel             : FILE;     { Datentyp fuer Aus- und Eingabe von bzw. zu den }
                              { Diskettenlaufwerken. }
 Buffer           : ARRAY [1..Sectorgroesse,1..Buffergroesse] OF byte;
    { In diesem Variablenfeld werden die zu kopierenden Daten gespeichert. }
 FCB              : ARRAY [0..25] OF byte absolute $005C;
 DMA              : ARRAY [0..255] OF byte;
    { Die letzten beiden Variablenfelder werden zum einlesen der Directory }
    { benoetigt. }
 Kopiernamen      : ARRAY [1..64] OF boolean;
    { Dieses Variablenfeld dient zur Speicherung der zu kopierenden Programme }
 Programme        : ARRAY [1..64,1..13] OF char;
    { Dieses Feld enthaehlt die Namen der Programme. }
 Irrtum,
 Wahr,
 Wahr2            : boolean; { Diese Variablen dienen dem Program zur Ueberpruefung }
                             { verschiedener Sachverhalte. }
 Quelldaten       : STRING [13]; { Variable zur Eingabe von Programmnamen }
 Laufwerk1,                      { Variable zur Eingabe dea aktuellen Laufwerks }
 Taste            : char;        { Variable, die zur Ueberpruefung von einem }
                                 { Tastendruck dient. }

 Fehler,
 Anfang,
 Sectoren,
 Merker           : integer;
 Schleife,
 Schleife2,
 Schleife3        : Schleifenvariablen; { Der Name sagt alles }

{$I copy.inc}  { Der zweite Teil des Programms wird von Diskette geladen. }
               { Dieser Teil muss unter dem Namen 'COPY.INC' vorhanden sind. }
{$I copy2.inc} { Dieser dritte wird des Programm wird auch von Diskette geladen }
               { Er muss unter dem Namen 'COPY2.INC' vorhanden sein. }

BEGIN { Hauptprogramm }
 REPEAT
  Auswahlmenu; { Das Hauptmenue wird ausgegeben. }
  Kopierverzweigung; { Wenn im Hauptmenu der Punkt Kopieren gewaehlt wurde, }
                     { springt das Programm zu dieser Prozedur }
 UNTIL 0=1; { Diese Wiederholungsschleife wird unendlich Durchlaufen. Ein }
            { Abbruch kann nur im Hauptmenu durchgefuehrt werden. }
END. { Ende des Hauptprogramms und somit auch des gesamten Kopierprogramms }
```

gramme von einer (oder auch von mehreren) Diskette(n) zu löschen. Als erstes fragt das Programm nach dem Laufwerk, in dem sich die Diskette mit dem zu löschendem Programm befindet. Nachdem die Frage beantwortet ist, interessiert sich das Programm nun für den Namen des Programms, das gelöscht werden soll. Nach der Eingabe des Namens erkundigt sich das Programm, ob das angegebene File auch wirklich gelöscht werden soll. Wird diese Frage bejaht, so wird das File gelöscht. Ansonsten wird dieses Unterprogramm abgebrochen und zum Hauptprogramm zurückgekehrt.

Der letzte Punkt im Hauptmenue dient zum Beenden des Programms.

Zuletzt noch einige Tips zum Abtippen und Aufstocken des Programms:

- Das Programm kann ohne Kommentare abgetippt werden.
- Das Programm kann nicht als Ganzes unter CP/M 2.2 im Speicher stehen. Erfahrungen unter CP/M 3.0 liegen mir nicht vor.
- Die Unterprogramme müssen unter den im Hauptprogramm angegebenen Namen auf der Diskette stehen.
- Um das Programm als COM-File auf Diskette zu schreiben, müssen Sie im Menuepunkt Compiler-Options von TURBO-PASCAL die 'C'-Taste drücken. Dies bewirkt, daß TURBO auf COM-File kompiliert. Durch Drücken der 'Q'-Taste kommen Sie wieder in das Hauptmenue von TURBO. Erneutes Drücken von 'C' verursacht, daß TURBO das Programm auf Diskette schreibt.
- Weitere Funktionen sind unter CP/M 2.0 nicht zu implementieren, da der Speicherplatz zu klein ist. Mehr Speicherplatz kann durch Verkleinern der Konstanten BUFFER erreicht werden. Dies verlangsamt jedoch den Kopierbetrieb. Bei einem Betrieb von TURBO mit erweitertem Speicherplatz, müßte sich durch Vergrößern der Variablen BUFFER die Kopiergeschwindigkeit steigern lassen.

Arndt Grass

```
PROCEDURE Dateneingabe;
{ Diese Prozedur zeigt die Namen der Programme auf Laufwerk A an und fragt, }
{ ob sie kopiert werden sollen. }
var Schleife4, Schleife5 : Schleifenvariablen;
BEGIN
 FOR Schleife4 := 1 TO Merker-1 DO { Loeschen der Variablen, in denen spaeter }
  Kopiernamen [Schleife4] := false; { gespeichert wird, was kopiert werden soll
}
 FOR Schleife4 := 1 TO Merker-1 DO
 BEGIN
  write ('   ');
  wahr2 := true;
  FOR Schleife5 := 1 TO 13 DO
   write (Programme [Schleife4,Schleife5]); { Anzeige des Programmnamens }
  write ('     ');
  REPEAT
   LowVideo;
   write ('[J/N] :'); { Kopieren Ja/Nein }
   NormVideo;
   read (KBD,Taste); { Abfrage Ja/Nein }
   Taste := UpCase(Taste); { Umwandlung in Grossbuchstaben }
   FOR Schleife5 := 1 TO 7 DO
    write (#08#16); { Loeschen von den letzten sieben Zeichen }
   IF (Taste<>'J') AND (Taste<>'N') THEN { Ist die Taste nicht J/N, dann }
   BEGIN
    LowVideo;
    write (' Fehler !!!!  Taste'); { schreibe eine Fehlermeldung, }
    NormVideo;
    write (#07); { gebe einen Klingelton aus, }
    read (KBD,Taste); { warte auf einen Tastendruck und }
    FOR Schleife5 := 1 TO 19 DO
     write (#08#16); { loesche die Fehlermeldung. }
    wahr2 := false;
```

```
    END
    ELSE
    BEGIN IF Taste = 'J' THEN { Ist die Taste 'J', dann }
          BEGIN
           Kopiernamen [Schleife4]:=true; { setze Kopiervariable }
           wahr2 := true;
           writeln ('J'); { und schreibe ein 'J' auf den Bildschirm }
          END
          ELSE
          BEGIN
           writeln ('N'); { ansonsten schreibe ein 'N' auf den Bildschirm }
           wahr2 := true;
           Kopiernamen [Schleife4] := false; { und setze die Kopiervariable zurue
ck }
          END;
      END;
   UNTIL wahr2; { wiederhole solange, bis Taste 'J' oder 'N' war. }
  END;
END;

PROCEDURE Auskunft;
{ Ausgabe einer Kurzanleitung zum Kopierprogram }
BEGIN
 ClrScr;
 writeln ('Die Quelldiskette muss sich in ''A'' ');
 writeln ('und die Zieldiskette muss sich in''B'' befinden !');
 writeln ('Bitte druecken Sie danach eine Taste');
 read (KBD,Taste);          { Wartet auf Tastendruck }
 BDOS($25,$0003);      { Setzt beide Laufwerke auf R/W }
 writeln;
END;

PROCEDURE Kopierverzweigung;
{ Diese Prozedur steuert den Kopierbetrieb }
BEGIN
 Auskunft; {Ausgabe der Kurzanleitung }
 Merker := 0;
 Katalog; { Die Prozedur zum Einlesen der Directory wird aufgerufen }
 Quelldaten:='';
 FOR Schleife := 1 TO 13 DO { Lesen des ersten Programmnamens }
  Quelldaten := Quelldaten + Programme [1,Schleife];
 IF Quelldaten <> '             ' THEN { Ist der Name nicht leer, dann }
 BEGIN
  Dateneingabe; { Die Prozedur, in der gefragt wird welche Programme kopiert }
                { werden sollen wird aufgerufen }
  FOR Schleife2 := 1 TO Merker-1 DO
   IF Kopiernamen [Schleife2] THEN Kopieren (Schleife2); { Die Kopier-
            { routine fuer die Programme, die kopiert werden sollen, wird aufge
rufen }
  Wiederholung; { Sollen weitere Diskette bearbeitet werden ? }
 END
 ELSE { Sonst zeige Fehlermeldung an }
 BEGIN
  LowVideo;
  writeln (#07,'Keine Datei');
  NormVideo;
  delay (2000);
 END;
END;
```

```
PROCEDURE Katalog;
{ Hole die Directory von der Diskette in Laufwerk 'A' }
 PROCEDURE HoleKatalog;
 { Diese Unterprozedur laedt die Namen der Programme in die Feldvariable }
 { Programme }
 BEGIN
  Merker := 1;
  Fehler := BDOS ($1a,Addr(DMA));
  FCB [0] := 0;
  FOR Schleife := 1 TO 11 DO
   FCB [Schleife] := Ord('?');
  Fehler := BDOS ($11,Addr(FCB));
  IF Fehler <> 255 THEN
  BEGIN
   Anfang := Fehler * 32;
   FOR Schleife := Anfang TO Anfang+8 DO
   Programme [Merker,Schleife-(Anfang-1)] := Char(Mem[Addr(DMA)+Schleife]);
   Programme [Merker,10] := '.';
   FOR Schleife := Anfang+9 TO Anfang+12 DO
    Programme [Merker,Schleife-(Anfang-2)] := Char(Mem[Addr(DMA)+Schleife]);
  END;
  REPEAT
   Merker := Merker+1;
   Fehler := BDOS ($12);
   Anfang := Fehler * 32;
   IF Fehler <> 255 THEN
   BEGIN
    FOR Schleife := Anfang TO Anfang+8 DO
     Programme [Merker,Schleife-(Anfang-1)] := Char(Mem[Addr(DMA)+Schleife]);
    Programme [Merker,10] := '.';
    FOR Schleife := Anfang+9 TO Anfang+12 DO
     Programme [Merker,Schleife-(Anfang-2)] := Char(Mem[Addr(DMA)+Schleife]);
   END
  UNTIL Fehler=255;
 END;

BEGIN
 FOR Schleife2 := 1 TO 64 DO   { Diese Schleifenkonstruktion loescht }
  FOR Schleife3 := 1 TO 13 DO  { Variable Programme }
   Programme [Schleife2,Schleife3] := ' ';
 HoleKatalog;  { Sprung zur Unterroutine }
END;

PROCEDURE Kopieren (zahl:integer);
{ Diese Prozedur uebernimmt die Nummer des Programms, das kopiert werden soll }
{ und kopiert es. }
BEGIN
 Quelldaten := ''; { Zuruecksetzen der Variablen }
 FOR Schleife3 := 2 TO 13 DO
  IF Ord(Programme [zahl,Schleife3])<>32 THEN
   IF (Programme [zahl,Schleife3]='.') OR ((Programme[zahl,Schleife3]>='0') AND
(Programme[zahl,Schleife3]<='9')) THEN
    Quelldaten := Quelldaten+Programme [zahl,Schleife3] { Umwandlung in Kleinbuc
hstaben }
   ELSE
    Quelldaten := Quelldaten+chr(Ord(UpCase(Programme [zahl,Schleife3]))+32);
 write (#07); { Klingelton ausgeben }
 write ('Kopiert wird: '); LowVideo; write (Quelldaten); NormVideo; writeln;
 { Anzeige des Programms, das gerade kopiert wird }
 assign(Quelle,'A:'+Quelldaten);
 assign(Ziel,'B:'+Quelldaten); { Anmelden der Dateien }
 reset(Quelle);
 rewrite(Ziel); { Oeffnen der Dateien }
 REPEAT
  blockread(Quelle,Buffer,Buffergroesse,Sectoren); { Lesen von der ersten Datei
}
  IF Sectoren <> 0 THEN { Beschreiben der zweiten Datei, wenn Bloecke }
   blockwrite(Ziel,Buffer,Sectoren); { gelesen werden konnten. }
 UNTIL Sectoren=0; { wiederholen bis kein Block mehr gelesen werden konnte.}
 close (Ziel);
 close (Quelle); { Schliessen der Dateien }
END;

PROCEDURE Loeschen;
```

```
{ Diese Prozedur dient zum Loeschen von einzelnen Programmen von einer }
{ Diskette in einem der beiden Laufwerke. }
VAR Fehler : integer;
BEGIN
 write ('Auf welchem Laufwerk soll geloescht werden ?  ');
 readln (Laufwerk1);
 Laufwerk1 := UpCase (Laufwerk1);
 if (Laufwerk1='A') or (Laufwerk1='B') then
 begin
  write ('Welches Programm soll geloescht werden ?  ');
  readln (Quelldaten);
  write ('Sind Sie sicher, dass Sie ',Quelldaten,' von Laufwerk ',Laufwerk1,' lo
eschen wollen ? ');
  read (KBD,Taste); { siehe oben }
  Taste := UpCase(Taste);
  IF Taste = 'J' THEN
  BEGIN
   LowVideo;
   writeln ('J');
   NormVideo;
   Wahr2 := true;
  END
  ELSE
  BEGIN
   LowVideo;
   writeln ('N');
   NormVideo;
   Wahr2 := false;
  END;
 END
 ELSE
  BEGIN
   Wahr2 := false;
   writeln(#07,'Fehler !!!');
  END;
 IF Wahr2 THEN   { Wenn geloescht werden soll, dann: }
 BEGIN
  BDOS($25,$0003);   { Beide Laufwerke werden auf R/W geschaltet. }
  {$I-}
  assign (Quelle,Laufwerk1+':'+Quelldaten);  { Der Zugriff auf das Programm wird
 angemeldet. }
  erase (Quelle);  { Das Programm wird von der Diskette geloescht. }
  Fehler := IOResult;
  {$I+}
  IF Fehler = $F1 then
   writeln ('Die Diskette ist voll');
  IF Fehler = 1 THEN
   writeln ('Dieses Programm existiert nicht !');
 END
 ELSE writeln (#07,' Abgebrochen !'); { Sonst wird diese Meldung ausgegeben. }
 delay (1000);
END; { Ende der Prozedur Loeschen }

PROCEDURE Auswahlmenu;
{ Diese Prozedur gibt das Menu des FASTCOPY aus. }
BEGIN
 REPEAT
  ClrScr;
  writeln ('                 FASTCOPY');
  writeln ('         (c) 1986  by Arndt Grass');
  writeln (' Herzlichen Dank an Brocken und Herrn Presche');
  writeln ('-------------------------------------------');
  writeln ('   Welches Unterprogramm wollen Sie benutzen ?');
  writeln ('                <K>opieren');
  writeln ('                <L>oeschen');
  writeln ('                <E>nde');
  write ('   Geben Sie ihre Wahl ein :');
  read (KBD,Taste);
  CASE Taste OF
     'k','K' : BEGIN  { Wenn Taste = 'K', dann }
                LowVideo;
                write ('K'); { Schreibe 'K' auf den Bildschirm }
                NormVideo;
                delay (1000); { Eine Sekunde lang warten. }
                writeln;
                Wahr := true; { Setzen der Variablen }
               END;
     'l','L' : BEGIN  { Wenn Taste = 'L', dann }
                LowVideo;
                write ('L'); { Schreibe 'L' auf den Bildschirm }
                NormVideo;
                writeln;
                delay (1000); { siehe oben }
                writeln;
                writeln;
                Loeschen;     { Sprung zum Unterprogramm Loeschen }
                Wahr := false;
               END;
     'e','E' : BEGIN
                LowVideo;
                write ('E'); { Schreibe 'E' auf den Bildschirm }
                NormVideo;
                writeln;
                delay (1000);
                BDOS ($00,$0000); { System-Neustart }
               END;
   ELSE { ansonsten }
    LowVideo;
    writeln (' Error'); { schreibe Fehlermeldung }
    NormVideo;
    write (#07);
    Wahr := false
   END;
  UNTIL Wahr; { Wiederhole bis Kopiert wersden soll }
  BDOS($25,$0003); { Setzen del Laufwerke auf R/W }
END;

PROCEDURE Wiederholung;
{ Diese Unterroutine fragt nach dem kopieren, ob eine Wiederholung erwuenscht }
{ ist. }
BEGIN
 Irrtum := false;
 REPEAT
  write ('Wollen Sie eine weitere Diskette bearbeiten ?  ');
  read (KBD,Taste);  { Der Zusatz KBD bedeutet das auf ein Zeichen von der }
                     { Tastatur gewartet wird. Das Zeichen muss nicht durch }
                     { 'ENTER' bestaetigt werden. }
  Taste := UpCase(Taste);   { Umwandlung in Grosszeichen }
  LowVideo;          { Bewirkt beim CPC eine Invertierung der Schrift }
  write (Taste);     { Das gelesene Zeichen wird auf den Bildschirm ausgegeben }
  NormVideo;         { Rueckkehr in die urspruengliche Schriftart }
  writeln;           { Naechste Zeile }
  IF Taste='J' THEN  { Ueberpruefung der Antwort }
   Irrtum := true;
  IF Taste='N' THEN
   BDOS ($00,$0000); { Wenn die Antwort Nein ist, dann wird ein Sprung ins }
                     { Betriebssystem durchgefuehrt. Dieser Sprung bewirkt }
                     { einen Reset des CP/M's. }
  IF NOT (Irrtum) THEN   { Ist die Eingabe falsch, so }
  BEGIN
   LowVideo;
   writeln (' FEHLER  !!!!!!!'); { wird eine Fehlermeldung ausgegeben und }
   NormVideo;
   write (#07);
   Irrtum := false;
  END;
 UNTIL Irrtum;        { die Abfrage wird wiederholt. }
END; { Ende der Prozedur Wiederholung }
```

JOYCE

Basic-Dateiverwaltung einf

Neben der Hardwareausstattung des JOYCE (z.B. kein Joystickanschluß) und dem Textverarbeitungsprogramm LocoScript, läßt auch der Befehlssatz des mitgelieferten Basic-Interpreters eindeutig erkennen, daß dieser Rechner mehr als »kleiner« Bürogehilfe, als zur Unterhaltung (spielen) gedacht ist. Es gibt z.B. keine Befehle zur Bildschirmsteuerung oder Tonerzeugung – dafür existiert eine Vielzahl von Funktionen zur Verwaltung von indizierten Dateien. Der Umgang mit solchen Dateien, den JETSAM-Funktionen des Mallard-Basic und der Unterschied zu anderen Datendateien, soll nun anhand eines einfachen Beispieles erläutert werden.

Versetzen wir uns in die Rolle eines Großhändlers, der die Adressen, Umsätze, Rechnungen usw. seiner Kunden mit Hilfe eines Computers verwalten möchte. Die einzelnen Daten werden als Datensätze in verschiedenen Dateien gehalten, z.B. Adresse in einer Adressdatei; Umsätze, Rechnungen usw. in entsprechenden Dateien. Um die Datensätze eines Kunden in den verschiedenen Dateien auffinden zu können, wird eine fortlaufende Kundennummer vergeben, über die diese Sätze eindeutig angesprochen werden können. Ein solcher Datensatz setzt sich wiederum aus mehreren Datenfeldern zusammen, z.B. könnte der Aufbau eines Datensatzes der Adressdatei mit fünf Feldern folgendermaßen aussehen:

Feld 1: Name
Feld 2: Vorname
Feld 3: Straße/Nr.
Feld 4: Postleitzahl
Feld 5: Ort

Im folgenden soll die Adressdatei als Beispiel und zur Demonstration der Unterschiede zwischen den verschiedenen Dateiarten dienen.

Am einfachsten wäre das Speichern der Adressen in einer sequentiellen Datei. Dieser Dateityp unterliegt aber einigen Einschränkungen:

Wie der Name schon sagt, werden hier die Datensätze der Reihe nach hintereinander in die Datei geschrieben und können nur in dieser Reihenfolge wieder gelesen werden. Benötigt man z.B. nur den 10. Datensatz (Kundennummer 10), so müssen die neun vorhergehenden Sätze überlesen werden – was natürlich Zeit in Anspruch nimmt. Noch gravierender ist die Tatsache, daß man in eine solche Datei nur schreiben oder nur aus ihr lesen kann. Änderungen sind also nicht ohne weiteres möglich – man muß die ganze Datei in den Speicher des Rechners einlesen, die Änderung durchführen und das Ganze dann wieder zurückschreiben. Dies hat zur Folge, daß die max. Größe unserer Adressdatei und damit die Anzahl der möglichen Kunden vom freien Speicher des Rechners abhängig ist. Andererseits befinden sich so alle Daten im Speicher des Rechners und der Zugriff auf einen Datensatz erfolgt wesentlich schneller als das Lesen von der Diskette.

Hier nun schematisch die nötigen Einzelheiten für die Adressdatei:

Da die Größe der Datei durch den freien Speicher im Rechner begrenzt ist und das Programm zur Verwaltung der Adressen auch etwas Speicherplatz benötigt, legen wir die max. Anzahl von Kunden mit z.B. 200 fest.

maxanz=200

Die Adressen (Datensätze) werden im Speicher des Rechners in einem zweidimensionalen Zeichenkettenfeld gespeichert, das entsprechend dimensioniert wird:

DIM adressen$(maxanz,5)

Über den ersten Index (1 bis **maxanz**) wird der Datensatz des gewünschten Kunden adressiert, er entspricht also der Kundennummer. Der zweite Index (1 bis 5) adressiert die einzelnen Datenfelder des Satzes (s.o.).

name$=adressen$(kundennummer,1)
vorname$=adressen$(kundennummer,2)
.
.
.
ort$=adressen$(kundennummer,5)

In einem Programm können nun alle nötigen Operationen (Eingeben, Ändern, Löschen) durch entsprechende Indizierung der Datensätze und Datenfelder vorgenommen werden.

Zum Speichern der Datei muß diese vorher eröffnet werden:

OPEN "O", #1,"ADRESSEN.SEQ"

Das **"O" (OUTPUT)** teilt Basic mit, daß die Datei ADRESSEN.SEQ geschrieben werden soll, wobei alle schon existierenden Dateien mit dem gleichen Namen gelöscht werden. Mit der Dateinummer **#1** wird eine logische Verbindung zwischen der Datei und den Ein-/Ausgabebefehlen hergestellt. Somit weiß Basic, wenn es z.B. den Befehl



PRINT # 1,"Meier" bearbeitet, daß "Meier" in die Datei ADRESSEN.SEQ geschrieben werden soll. Hätten wir noch eine zweite Datei mit **OPEN "O",#2,"TEXT"** eröffnet, so würde mit **PRINT # 2,"Meier"** der Text in diese Datei geschrieben. Basic erlaubt für die Dateinummer eine Zahl von 1 bis 3, es können also max. drei Dateien gleichzeitig bearbeitet werden. Diese Anzahl kann aber mit Hilfe des **MEMORY**-Befehls geändert werden, falls eine größere Anzahl gleichzeitig offener Dateien notwendig sein sollte.

Das eigentliche Schreiben der Datensätze geschieht durch:

```
FOR i=1 to maxanz
    FOR j=1 to 5
        PRINT # 1,adressen$(i,j)
    NEXT
NEXT
```

Nachdem alle Daten geschrieben worden sind, muß die Datei mit

CLOSE # 1

wieder geschlossen werden. Geschieht dies nicht, so können Daten verloren gehen und die Datei kann nicht zum Lesen geöffnet werden.

Zum Einlesen einer solchen Datei wird diese mit

OPEN "I", #1,"ADRESSEN.SEQ"

geöffnet. Das "I" steht hier für Eingabe **(Input)** und die Daten können durch

```
FOR i=1 to maxanz
    FOR j=1 to 5
        INPUT #1,adressen$(i,j)
    NEXT
NEXT
```

gelesen werden.
Danach wird die Datei ebenfalls mit **CLOSE # 1** wieder geschlossen.

Damit wären alle notwendigen Operationen zur Realisierung der Adressdatei als sequentielle Datei vorhanden. Da wir uns aber in die Rolle eines Großhändlers versetzt haben, werden die max. 200 Adressen, die auf diese Art verwaltet werden können, nicht genügen.

Diese Einschränkung läßt sich aber durch Verwendung von Direktzugriffsdateien (engl. Random Access) umgehen. Dieser Dateityp hat gegenüber der sequentiellen Datei den wesentlichen Vorteil, daß die Daten in jeder beliebigen Reihenfolge geschrieben und gelesen werden können. Dadurch ist die Größe einer solchen Datei nur von der Kapazität des Speichermediums (Diskette, Festplatte) bzw. der max. Dateigröße, die das Betriebssystem des Rechners unterstützt, begrenzt. Auch können Änderungen leicht vorgenommen werden – es wird nur die Adresse des betroffenen Kunden in den Speicher gelesen, geändert und wieder an die gleiche Stelle in die Datei zurückgeschrieben. Dies ist möglich, da in einer Direktzugriffsdatei die Daten völlig anders als in einer sequentiellen Datei gespeichert werden. Der Inhalt der sequentiellen Adressdatei könnte z.B. folgendes Aussehen haben:

Meier
Hans
Im Eck 123
4321
Tünneskirchen
Schmitd
Rudolf
Hinterm Mond 7 - 9
4711 Dingsda
Büchsenmacher
Bimbo
...usw.

Die Datenfelder der Datensätze sind als Zeichenketten unterschiedlicher Länge mit einem von Basic angefügten Zeilenende, das beim Lesen wieder als Trennzeichen der Felder dient, geschrieben worden. Daraus ergeben sich für die einzelnen Datensätze ebenfalls unterschiedliche Längen und man kann leicht erkennen, daß so ein direkter Zugriff auf ein bestimmtes Datum nicht möglich ist.

Dagegen haben die Datensätze in einer Direktzugriffsdatei eine feste Länge, die entsprechend der Größe der zu speichernden Informationen festgelegt wird, sowie eine Satznummer, über die die Sätze angesprochen werden. Die Größe der einzelnen Datenfelder eines solchen Satzes ist ebenfalls konstant – man muß sich vorher überlegen, wieviele Zeichen z.B. max. für den Namen eines Kunden benötigt werden. Aus der Summe der einzelnen Feldlängen ergibt sich dann die Gesamtlänge (Satzlänge) eines Datensatzes in Bytes.

Für die Adressdatei könnte das so aussehen:

Name:	20 Zeichen
Vorname:	15 Zeichen
Straße:	25 Zeichen
PLZ:	4 Zeichen
Ort:	24 Zeichen
Summe:	89 Zeichen (Bytes)

satzlänge=89

Um mit einer Direktzugriffsdatei arbeiten zu können, muß diese ebenfalls eröffnet werden:

OPEN "R", #1,"ADRESSEN.RA", satzlänge

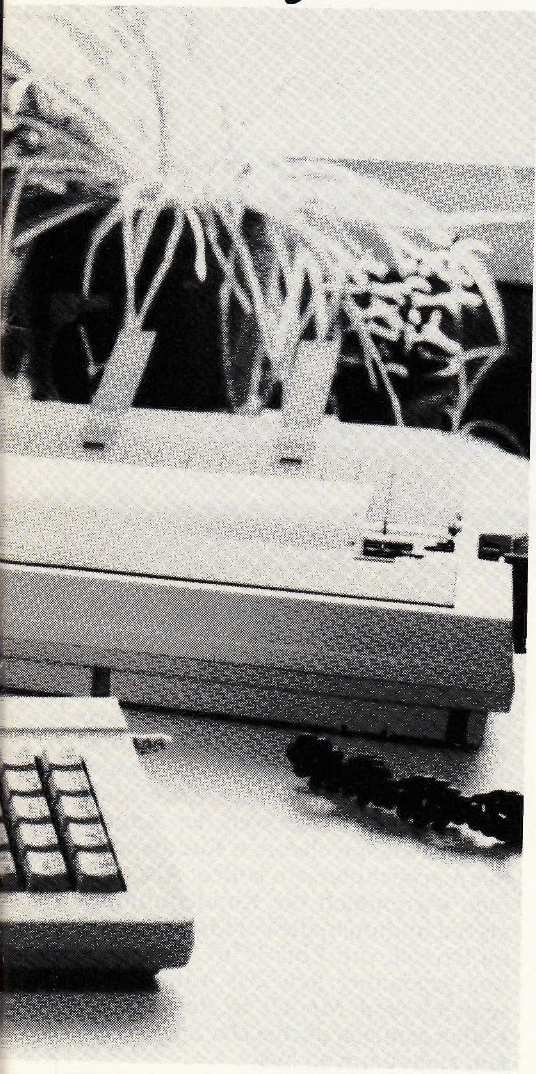
Hier zeigt das **"R"** an, daß die Adressdatei als Direktzugriffsdatei (Random Access) eröffnet werden soll. Ein weiterer Unterschied zum Eröffnen einer sequentiellen Datei, ist die zusätzliche Angabe der Satzlänge von 89 Bytes hinter dem Dateinamen, die wir oben aus den benötigten Größen der Datenfelder errechnet hatten. Anhand dieser Größe und der Satznummer (s.u.) kann die Position des Datensatzes in der Datei von Basic berechnet werden. Wird diese Angabe weggelassen, so nimmt Basic eine Satzlänge von 128 Bytes an, was in diesem Fall eine Verschwendung von 39 Bytes pro Adresse bedeuten würde. Es können aber auch Sätze mit mehr als 128 Bytes Größe verarbeitet werden.

Dazu muß allerdings auch die Größe des Zwischenspeichers (Satzbuffer), in dem ein Datensatz aus der Datei gelesen bzw. aus dem in die Datei geschrieben wird, mit Hilfe des **MEMORY**-Befehls angepaßt werden, da Basic hierfür ebenfalls 128 Bytes vorsieht.

Beim Eröffnen einer Direktzugriffsdatei wird eine schon vorhandene Datei im Gegensatz zur sequentiellen Datei nicht gelöscht. Daher können Daten verändert oder hinzugefügt werden. Existiert die Datei noch nicht, so entsteht eine neue, vorläufig leere Datei.

Als nächstes muß die Struktur des Satzpuffers und somit die der Datensätze definiert werden, d.h. die Anzahl und Größen der einzelnen Datenfelder. Dies geschieht mit der **FIELD**-Anwei-

...ach mit Jetsam



sung, wodurch die darin aufgeführten Feldvariablen dem Satzpuffer der Datei zugeordnet werden. Für die Adressdatei sieht das folgendermaßen aus:

FIELD # 1,20 AS name$,
15 AS vorname$,
25 AS straße$,
4 AS plz$,
25 AS ort$

Bei den Feldvariablen handelt es sich immer um Zeichenkettenvariablen. Die Längenangaben **n AS** bestimmen die Größe der Felder in Bytes (Zeichen). Die Reihenfolge der Felder im Satzpuffer ist durch die Anordnung der Feldvariablen in der **FIELD**-Anweisung bestimmt. Bei der obigen Struktur für die Adressdatei werden die ersten 20 Bytes durch die Zeichenkette **name$** belegt, die nächsten 15 Bytes von **vorname$** usw.

RSET name$="Meier"

der Name rechtsbündig in das Feld eingetragen, wobei das Feld von links mit Leerzeichen aufgefüllt wird. Schließlich werden mit

MID$(name$,10,3)="Meier"

nur die ersten drei Zeichen, also **Mei**, ab dem zehnten Byte in das Feld gespeichert, wobei der restliche Inhalt unverändert bleibt.

In allen drei Fällen werden für die Feldvariablen zu lange Zeichenketten bei der Zuweisung auf die Größe der Feldvariablen gekürzt.

Wird dagegen eine Zuweisung wie
name$="Meier"
oder
INPUT name$

durchgeführt, so wird die Zugehörigkeit der Feldvariablen **name$** zum Satzpuffer der Direktzugriffsdatei aufgehoben und sie kann als normale Zeichenkette weiter verwendet werden.

sprechen wieder den Datenfeldern **name$, vorname$,** usw.

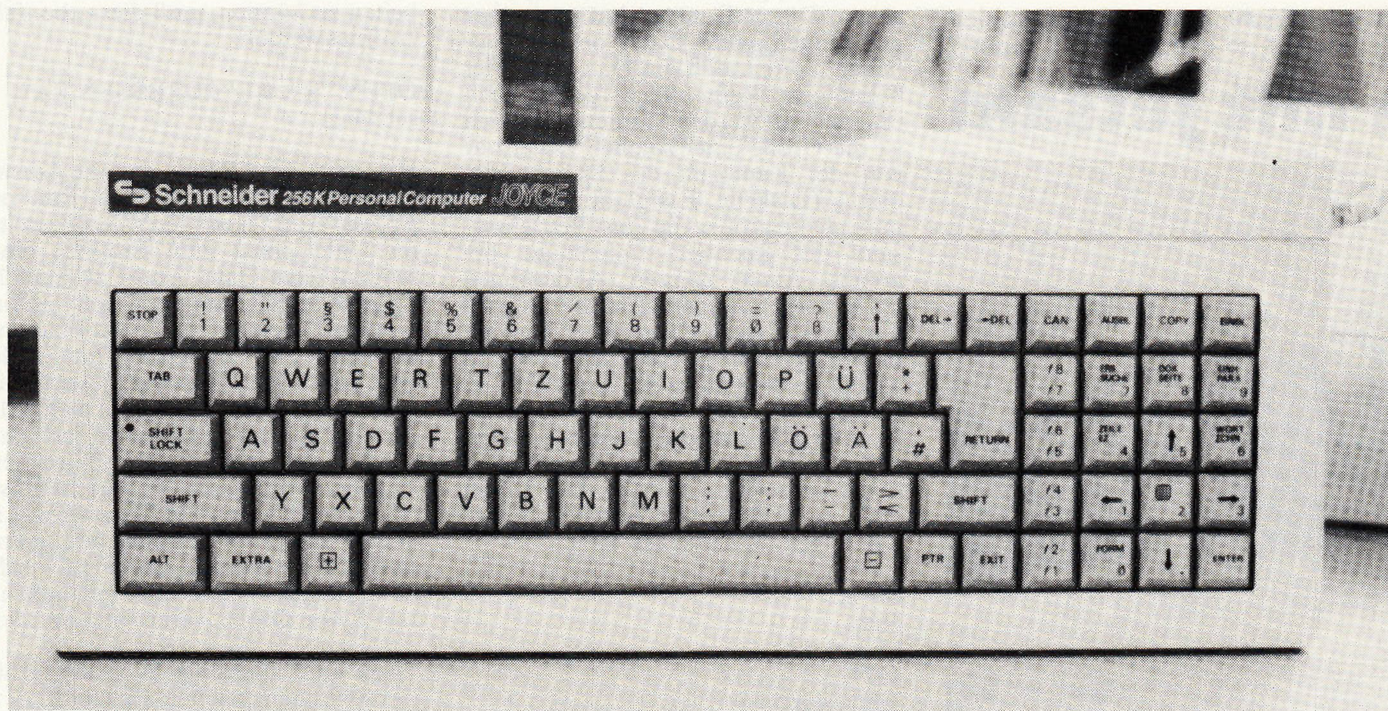
Das Füllen des Satzpuffers mit den in **adresse$()** stehenden Daten geschieht dann mit den Befehlen

LSET name$=adress$(1)
LSET vorname$=adress$(2)
LSET straße$=adress$(3)
LSET plz$=adress$(4)
LSET ort$=adress$(5)

Nun kann die in dem Satzpuffer enthaltene Adresse als Datensatz in die Direktzugriffsdatei geschrieben werden. Dafür ist der **PUT**-Befehl zuständig, der auf zwei verschiedene Weisen angewendet werden kann. Mit

PUT # 1

werden die Datensätze, wie in einer sequentiellen Datei, hintereinander geschrieben. Nach jedem **PUT # 1** wird die Satznummer automatisch um eins erhöht und steht somit auf dem nächsten zu schreibenden/lesenden Datensatz. Direkt nach dem Eröffnen der Datei wären das die Sätze 1, 2, 3 usw. Hat man aber schon z.B. den fünften Satz aus der Datei gelesen, so wären das die Sätze 6, 7, 8 etc.



Bevor man nun einen Datensatz in eine Direktzugriffsdatei schreiben kann, muß der Satzpuffer mit den zu schreibenden Werten durch Zuweisungen an die Feldvariablen gefüllt werden.
Dazu stehen drei verschiedene Zuweisungsarten zur Verfügung, deren Wirkungsweise anhand der Feldvariablen **name$** gezeigt wird:

LSET name$="Meier"

Mit dieser Zuweisung wird der Name **Meier** linksbündig in die Feldvariable **name$** geschrieben und die restlichen im Feld unbenutzten Bytes mit Leerzeichen aufgefüllt. Entsprechend wird durch

Daher ist es ratsam, die zu speichernden Daten zur Manipulation (Eingaben, Änderungen,...) in normalen Variablen zu halten und deren endgültigen Inhalt erst zum Speichern den entsprechenden Feldvariablen zuzuweisen. Dazu bietet sich das, von der sequentiellen Adressdatei schon bekannte, Zeichenkettenfeld an, das jetzt aber nur noch eine, nämlich die gerade zu bearbeitende Adresse aufnehmen muß:

DIM adresse$(5)

Die fünf Elemente dieses Feldes ent-

Mit der zweiten Form kann man dagegen die Satznummer bestimmen, unter der der Datensatz gespeichert werden soll. Dies ist für unsere Adressdatei wichtig, da die Adressen der Kunden über die entsprechenden Kundennummern erreichbar sein sollen. Also nehmen wir einfach die Kundennummer als Satznummer, die bei Mallard-Basic im Bereich von 1 bis 32767 liegen darf, was gegenüber den 200 Adressen im sequentiellen Modell in den meisten Fällen genügen dürfte. Der Speicherbefehl sieht dann so aus:

PUT # 1,kundennummer

Es sei noch erwähnt, daß vor jedem Schreiben eines Datensatzes der Satzpuffer mit den entsprechenden Daten (hier die Adressen) neu gefüllt werden muß! Anderenfalls wird bei jedem PUT-Befehl die gleiche Adresse unter der angegebenen Kundennummer gespeichert.

Für das Lesen eines Satzes in den Satzpuffer ist der **GET**-Befehl zuständig.

Dieser Befehl kann ebenfalls in zwei Varianten angewandt werden, ohne oder mit Angabe der Satznummer. Er verhält sich genau wie der **PUT**-Befehl, mit dem Unterschied, daß die Daten gelesen werden und daß bei dem Versuch, einen noch nicht geschriebenen Datensatz zu lesen, ein Fehler entsteht. Für die Adressdatei ist wieder, wie beim Schreiben, die zweite Art notwendig:

GET # 1,kundennummer

Nach diesem Befehl befindet sich die Adresse des gewünschten Kunden im Satzpuffer und der Inhalt der Datenfelder kann über die Feldvariablen angesprochen werden. Z.B. kann man mit dem Befehl

PRINT name$,vorname$,...

die Adresse auf den Bildschirm ausgeben oder ihren Inhalt wieder in das Zeichenkettenfeld **adress$()** mit

adress$(1)=name$
adress$(2)=vorname$
.
.
adress$(5)=ort$

zu weiteren Manipulationen übertragen. Dort kann jetzt z.B. eine Adressänderung vorgenommen werden und die neue Adresse wieder unter der gleichen Kundennummer in die Datei zurückgeschrieben werden.

Wie sequentielle Dateien, müssen auch Direktzugriffsdateien nach Beendigung der Arbeit wieder mit **CLOSE # Dateinummer** geschlossen werden.

Mit Hilfe der Direktzugriffsdatei konnte, gegenüber der sequentiellen Datei, die Anzahl der zu verwaltenden Kunden erheblich gesteigert werden. In beiden Fällen können wir die Adressen der Kunden aber nur über die Kundennummer ansprechen. Möchten wir dagegen wissen, welche Kunden z.B. in einer bestimmten Stadt bzw. in einem bestimmten Postleitzahlengebiet seßhaft sind, so müssen alle Adressen auf die Erfüllung der Suchbedingungen untersucht werden. Ebenso problematisch ist die Erstellung einer Adressliste, die alphabetisch nach dem Namen sortiert ist.

Zu diesem Zweck wäre ein alphabetisches Verzeichnis der Namen nützlich, in dem diese und die dazugehörigen Satznummern, unter der die vollständige Adresse in der Direktzugriffsdatei gespeichert ist, enthalten sind. Dateien, die mit Hilfe solcher **Index**-Tabellen verwaltet werden, nennen sich **indizierte** oder **indexsequentielle** Dateien.

Dabei handelt es sich um Direktzugriffsdateien, zu denen zusätzlich eine **Indexdatei** angelegt wird. Beim Speichern eines neuen Datensatzes in die Direktzugriffsdatei wird ein **Schlüsselbegriff** angegeben, der von der Dateiverwaltung zusammen mit der Satznummer des Datensatzes in die Indexdatei entspechend einsortiert wird. Zum Suchen eines Datensatzes kann dann dieser Schlüsselbegriff wieder verwendet werden und falls der Schlüssel im Index gefunden wurde, mit Hilfe der Satznummer, der vollständige Datensatz wieder gelesen werden. Die Adressdatei, mit Hilfe der indizierten Dateiverwaltung, könnte so aussehen, wobei die Satznummer wieder der Kundennummer entspricht:

Direktzugriffsdatei:

Satz 1: Meier
Hans
Im Eck 123
4321
Tünneskirchen

JOYCE

Satz 2: Schmidt
Rudolf
Hinterm Mond 7 – 9
4711
Dingsda

Satz 3: Büchsenmacher
Bimbo
Holzweg 1
4545
Irgendwo

Indexdatei mit sortierten Namen:

Büchsenmacher, Satz 3
Meier, Satz 1
Schmidt, Satz 2

oder Indexdatei mit sortierter PLZ:

4321, Satz 1
4545, Satz 3
4711, Satz 2

Die meisten Basic-Dialekte, die bei Heim- und Personalcomputern zu finden sind, unterstützen die indizierte Dateiverwaltung leider nicht. In dem Fall bleibt es dem Programmierer überlassen, diese zu verwirklichen. Mallard-Basic bietet dagegen mit den **JETSAM**-Funktionen sehr komfortable Werkzeuge zum Arbeiten mit diesem Dateityp.

Im Gegensatz zur Direktzugriffsdatei wird eine indizierte Datei beim Öffnen nicht automatisch erstellt. Vor der ersten Benutzung muß diese erst mit dem **CREATE**-Befehl eingerichtet und initialisiert werden. Für unsere Adressdatei geschieht das durch

```
CREATE # 1,"ADRESSEN.DAT"
       "ADRESSEN.INX"
       2,satzlänge
```

Die Dateinummer **#1** und ihre Bedeutung ist ja schon von der sequentiellen und der Direktzugriffsdatei bekannt.
Der Dateiname **ADRESSEN.DAT** bezeichnet die Datendatei, in der die vollständigen Datensätze (Adressen) der Kunden gespeichert werden. Dabei handelt es sich um eine ganz normale Direktzugriffsdatei, mit den oben schon beschriebenen Eigenschaften.

Der neu hinzugekommene zweite Dateiname **ADRESSEN.INX** bezeichnet die von Jetsam einzurichtende Indexdatei, in der die Schlüsselwörter mit den dazugehörigen Satznummern der Datendatei von Jetsam eingetragen und verwaltet werden. Für diese Datei benötigt Jetsam mindestens einen Zwischenspeicher (engl. Buffer), über den es die zur Verwaltung des Index nötigen Operationen abwickeln kann. Bei erstmaligem Laden von Basic existiert dieser noch nicht, da er einen Speicherplatz von 128 Bytes belegt, der vom verfügbaren Platz für Programme abgezogen wird und bei Nichtverwendung von Jetsam unnötig ist. **Vor** der ersten Verwendung von Jetsam in einem Programm muß dieser Zwischenspeicher mit

```
BUFFERS anzahl
```

eingerichtet werden. **anzahl** kann im Bereich von 0 bis 255 liegen und gibt an, wieviele Index-Puffer (jeder 128 Bytes groß) reserviert werden sollen. Je mehr Buffer Jetsam zur Verfügung gestellt werden, umso schneller erfolgt die Bearbeitung der indizierten Dateien.

Neu ist auch der Parameter **2**, der die Datei für andere Benutzer in einem Mehrbenutzersystem sperrt. Auf dem Joyce ist dieser Parameter uninteressant, muß aber aus Kompatibilitätsgründen angegeben werden. Dieser **Sperr-Parameter** tritt auch in anderen Jetsam-Funktionen auf, wo dann ebenfalls konstante Werte benutzt werden.

Die Bedeutung der **satzlänge** ist ebenfalls schon bekannt – sie gibt wieder die Länge eines Datensatzes in der Datendatei an. Allerdings benötigt Jetsam in jedem Satz zur Verwaltung der Datendatei die ersten zwei Bytes und reserviert sich diese; dem Benutzer stehen sie also nicht zur Verfügung. Da wir für eine Adresse 89 Bytes benötigen, muß im Gegensatz zu normalen Direktzugriffsdateien, die Satzlänge um zwei erhöht werden:

```
satzlänge=89+2
```

Wenn vom **CREATE**-Befehl beide Dateien erfolgreich erzeugt wurden, steht die indizierte Datei zur weiteren Bearbeitung bereit, man braucht sie also nicht mehr, wie folgend beschrieben, eröffnen.

Existiert dagegen die Adressdatei schon, dann wird mit

```
OPEN "K", # 1,"ADRESSEN.DAT"
         "ADRESSEN.INX
         2,satzlänge
```

die indizierte Datei zur Bearbeitung eröffnet.

Die Parameter ab der Dateinummer **#1** entsprechen denen des **CREATE**-Befehls. Der erste Parameter **"K"** zeigt Basic, daß eine existierende, indizierte Datei geöffnet werden soll.

Nach dem **CREATE**- bzw. dem **OPEN**-Befehl, muß nun wieder, wie bei der einfachen Direktzugriffsdatei, die Struktur des Satzpuffers definiert werden. Dabei muß die Summe der Feldlängen um zwei Bytes kleiner sein als die bei **CREATE/OPEN** angegebene Satzlänge, da Jetsam, wie erwähnt, zwei Bytes für sich in Anspruch nimmt. Die **FIELD**-Anweisung entspricht exakt der für die Direktzugriffsdatei benutzten:

```
FIELD # 1,20 AS name$,
          15 AS vorname$,
          25 AS straße$,
           4 AS plz$,
          25 AS ort$
```

Ebenfalls benutzen wir zur Manipulation einer Adresse wieder das Zeichenkettenfeld **adress$()**, um zu vermeiden, daß durch eine falsche Zuweisung die Zugehörigkeit einer Feldvariablen zum Satzpuffer aufgehoben wird.

Um nun mit indizierten Dateien sinnvoll arbeiten zu können, muß überlegt werden, nach welchen Kriterien die gespeicherten Informationen wieder gesucht oder Listen erstellt werden sollen.

Innerhalb der Indexdatei kann Jetsam bis zu acht verschiedene Indextabellen verwalten. D.h. für unsere Adressdatei, daß wir gleichzeitig ein sortiertes Verzeichnis der Namen und eines der Postleitzahlen benutzen können. Theoretisch könnten für alle fünf Datenfelder der Adressdatei entsprechende Indextabellen angelegt werden, was aber wenig Sinn ergibt, da dadurch mehr Speicherplatz verbraucht wird und wir kaum nach irgendwelchen Straßen suchen werden.

Ein Problem stellt sich durch die von uns vorgeschriebene und zu jeder Adresse gehörende Kundennummer ein. In der sequentiellen Datei entsprach diese dem Index im Zeichenkettenfeld **adress$()** und in der Direktzugriffsdatei der Satznummer des Datensatzes mit der Adresse des Kunden. Die Daten eines Kunden wurden also direkt über die Kundennummer angesprochen. Dies ist jetzt bei der Verwendung einer indizierten Datei nicht mehr möglich, da Jetsam selbständig die Satznummer bestimmt, unter der ein Datensatz in die Datei geschrieben wird. Dadurch ergibt sich die Notwendigkeit, eine Indextabelle für die Kundennummer einzusetzen.

Für unsere Adressdatei benutzen wir also drei Indextabellen; eine für die Kundennummer (Tabelle 0), eine für die Namen (Tabelle 1) und eine für die Postleitzahlen (Tabelle 2). Jede einzelne der acht möglichen Tabellen wird in den Jetsam-Funktionen durch ihre Nummer selektiert, die zwischen 0 und 7 liegen darf. Um später bei der Verwendung der Jetsam-Funktionen den Parameter, der die einzelnen Indextabellen selektiert, besser erkennen zu können, verwenden wir die Konstanten **kindex=0** für die Tabelle der Kundennummern, **nindex=1** für die der Namen und **pindex=2** für die der Postleitzahlen.

Des weiteren muß überlegt werden, ob in einem Index ein Schlüsselwert nur einmal oder mehrfach vorkommen darf. Bei der Namen- und Postleitzahlentabelle ist ein mehrfaches Auftreten

eines Wertes sicher sinnvoll, da es, wie allgemein bekannt, unzählige Meiers, Müllers usw. gibt. Ebenfalls ist nicht auszuschließen, daß mehrere Kunden in derselben Stadt residieren. Dagegen darf eine Kundennummer nur einmal vorhanden sein, da sonst die Daten aus den einzelnen Dateien (s. ganz oben) nicht eindeutig einander zuzuordnen sind.

Deshalb bietet Jetsam die Möglichkeit, für jede einzelne Tabelle ein einfaches oder mehrfaches Vorkommen eines Schlüsselwertes zu erlauben. Beim Erzeugen einer indizierten Datei mit **CREATE**, wird für alle acht möglichen Indextabellen automatisch ein mehrfaches Auftreten eines Wertes erlaubt. Soll aber in bestimmten Tabellen ein Wert nur einmalig vorkommen dürfen, so muß diese Eigenschaft den betroffenen Tabellen mitgeteilt werden. Die Jetsam-Funktion dafür ist

ergebnis=RANKSPEC(dateinummer,
Tabellennummer,
attribut)

Mit **attribut=0** wird ein Schlüsselwert mehrfach in der entsprechenden Indextabelle erlaubt. Dies kann man sich ersparen, da diese Einstellung automatisch angenommen wird. Mit **attribut=1** darf ein Wert in der Tabelle nur einmalig enthalten sein. Als Ergebnis liefert die Funktion den Wert Null, wenn kein Fehler aufgetreten ist.

Diese Einstellungen müssen (wenn nötig) bei der Erstellung einer indizierten Datei mit **CREATE** vorgenommen werden, bevor Daten gespeichert werden und bleiben für die erzeugte Datei bis zu ihrem Ende durch Löschen aktiv, da diese Information in der Indexdatei mitgespeichert wird und nach einem Wiedereröffnen mit **OPEN** in Kraft tritt.
Um also in der Adressdatei kein mehrfaches Vorkommen einer Kundennummer zu erlauben, muß diese nach dem **CREATE**-Befehl mit

ergebnis=RANKSPEC(#1,kindex,1)

Jetsam mitgeteilt werden.

Nach diesen ganzen Vorüberlegungen und Maßnahmen können nun endlich Adressen in die indizierte Datei eingetragen werden.

Wie bei der normalen Direktzugriffsdatei, müssen dazu die Adressdaten erst in den Satzpuffer übertragen werden:

LSET name$=adress$(1)
LSET vorname$=adress$(2)
usw.

Das Speichern des Datensatzes geschieht jetzt nicht mehr mit dem **PUT**-Befehl, sondern durch die Jetsam-Funktion **ADDREC**. Diese Funktion schreibt den Satz, unter einer von Jetsam vergebenen Satznummer, in die Datendatei und sortiert gleichzeitig den angegebenen Schlüsselwert in die angegebene Indextabelle ein. Hier der vollständige Aufruf der Funktion:

ergebnis=ADDREC(Dateinummer,
Sperrparameter,
Tabellennummer,
Schlüsselwert)

Die Bedeutung der Datei- und Tabellennummer dürfte aus obigen Ausführungen schon bekannt sein. Wie bei **CREATE** und **OPEN**, muß der Sperrparameter wieder aus Kompatibilitätsgründen zur Mehrbenutzerversion angegeben werden und hat ebenfalls den Wert 2.

Beim Schlüsselwert handelt es sich immer um eine Zeichenkette mit einer maximalen Länge von 31 Zeichen. Für die Adressdatei nehmen wir als Schlüsselwert die Kundennummer und als Tabellennummer die Konstante **kindex** (Tabelle 0, in der ein Wert nur einmal existieren darf).

Da Basic Zahlen intern nicht als Ziffernfolge, sondern in einer, für Computer geeigneteren Form darstellt, muß die Kundennummer vor Verwendung als Schlüsselwert einer Jetsam-Funktion in eine Zeichenkette umgewandelt werden. Dazu gibt es einmal die **STR$**-Funktion, die eine Zahl von der internen Darstellung wieder in eine Ziffernfolge umwandelt:

schlüssel$=STR$(kundennummer)

Diese Form hat aber den Nachteil, daß für jede Ziffer der Zahl ein Byte benötigt wird, während in der internen Darstellung von Basic nur zwei Bytes nötig sind, um Zahlen im Bereich von −32768 bis +32767 darzustellen.

Außerdem entsteht in der Indextabelle nicht die erwartete, aufsteigende Reihenfolge der Kundennummern 1, 2, 3,..., 10, usw., da bei Zeichenketten ein Vergleich auf kleiner-, gleich und größer als, anders als bei Zahlen ausgeführt wird. In der Indextabelle würde sich deshalb folgende Reihenfolge einstellen: 1, 10, 2, 3,...

Deshalb bietet Mallard-Basic noch die zwei Funktionen **MKIK$** und **MKUK$** an, die beide aus der internen Darstellung einer Zahl eine zwei Bytes lange Zeichenkette erzeugen, die als Schlüsselwert für Jetsam-Funktionen verwendet werden kann. Mit **MKIK$** können dabei ganze Zahlen im Bereich von −32768 bis +32767 und mit **MKUK$** ganze Zahlen in einem Bereich von 0 bis 65535 umgewandelt werden. Diese so erzeugten Zeichenketten lassen sich dann mit den Funktionen **CVIK** und **CVUK** wieder in die, für Basic zum Rechnen nötige, interne Darstellung zurückwandeln. Dies ist auch nötig, wenn man den Wert z.B. drucken möchte. Die Ausgabe der Zeichenkette selbst erzeugt nur wirre Zeichen auf dem Papier!

Der Befehl zum Hinzufügen eines neuen Kunden sieht also so aus:

ergebnis=ADDREC(#1,
2,
kindex,
MKUK$
(kundennr.))

Wird als Ergebnis der Wert 0 zurückgegeben, so konnte die Adresse erfolgreich in die Datendatei, und die Kundennummer in die Tabelle der Indexdatei, eingetragen werden. Wenn versucht wird, eine Adresse unter einer in der Tabelle schon existierenden Kundennummer hinzuzufügen, was bei der Erstellung der Datei mit **RANKSPEC**



»verboten« wurde, dann wird in **ergebnis** der Fehlercode 116 zurückgegeben und die Adresse nicht gespeichert.
In beiden Fällen wird von Jetsam eine **aktuelle Position** festgehalten. Diese Position bestimmt bei einigen Jetsam-Funktionen, auf welchen Datensatz, welcher Indextabelle und welchem Schlüsselwert sich die Funktion auswirkt. Einige Funktionen verlangen diese Position unbedingt als Eingabeparameter, bei anderen kann sie optional angegeben werden. Die Position setzt sich aus den drei Werten **Tabellennummer**, **Schlüsselwert** und **Satznummer** zusammen. Im Falle des Erfolges bei **ADDREC** zeigt diese auf die gerade hinzugefügte Adresse, bei Fehler 116 auf die schon existierende. Bei anderen Funktionen kann die Position nach Auftreten von Fehlern auch undefiniert sein. Um die aktuelle Position für Programme verfügbar zu machen, bietet Jetsam folgende drei Funktionen:

Tabellennr.=FETCHRANK(dateinr.)
Schlüssel$=FETCHKEY$(dateinr.)
Satznr.=FETCHREC(dateinr.)

Nun haben wir aber erst die gleiche Möglichkeit zum Lesen der Adresse aus der indizierten Datei, wie auch bei der sequentiellen und der einfachen Direktzugriffsdatei – die Kundennummer. Deshalb gibt es bei Jetsam die **ADDKEY**-Funktion, die das Hinzufügen weiterer Schlüssel für einen, in der Datendatei bereits gespeicherten, Satz erlaubt:

ergebnis=ADDKEY(Dateinummer,
Sperrparameter,
Tabellennummer,
Schlüsselwert,
Satznummer)

Hierbei muß für den Sperrparameter der Wert 0 angegeben werden. Für das Ergebnis der Funktion gilt das gleiche wie bei **ADDREC**, jedoch mit dem Unterschied, daß der betroffene Datensatz schon existiert.
Im Falle, daß bei **ADDREC** kein Fehler auftrat, können nun der Name und die Postleitzahl in die entsprechenden Indextabellen einsortiert werden. Da die **ADDKEY**-Funktion dazu die Satznummer wissen muß, unter der die Adresse in der Datendatei zu finden ist, müssen wir uns diesen mit

satznr.=FETCHREC(#1)

von Jetsam geben lassen.
Das Einsortieren des Namens geschieht dann mit:

ergebnis=ADDKEY(#1,0,nindex,
adress$(1),satznr.)

Das Einsortieren der Postleitzahl in die Tabelle **pindex** stellt uns nun wieder vor ein Problem. Wenn man davon ausgeht, daß die Postleitzahl immer mit vier Ziffern in **adress$(4)** eingegeben wird, dann erhält man bei Verwendung dieser Zeichenkette als Schlüsselwert eine numerisch richtig aufsteigend sortierte Indextabelle, anderenfalls treten die gleichen Probleme, wie bei der Kundennummer als Schlüsselwert, auf (s.o.). Alternativ kann die in **adress$(4)** als Zeichenkette dargestellte Postleitzahl vor der Verwendung der **MKIK$** bzw. **MKUK$** durch Anwendung der **VAL**-Funktion in die interne Darstellung einer Zahl gebracht werden. In dem Fall, muß die durch **FETCHKEY$** erhaltene Zeichenkette des Schlüsselwertes mit **CVIK/CVUK** wieder in eine Zahl und, wenn nötig, mit der **STR$**-Funktion wieder in eine normale Zeichenkette zurückgewandelt werden. Der Einfachheit halber entschieden wir uns für die erste Möglichkeit, also

ergebnis=ADDKEY(#1,0,pindex,
adress$(4),satznr.)

Beim Einsortieren dieser beiden Schlüsselwerte kann von der **ADDKEY**-Funktion als Ergebnis nur der Wert Null zurückgegeben werden, da in den entsprechenden Indextabellen ein mehrfaches Auftreten eines Schlüssels zugelassen ist.

Damit wären für das Eintragen neuer Adressen und ihrer Schlüsselwerte in die indizierte Datei alle notwendigen Schritte erläutert – wenden wir uns nun dem Wiederauffinden der gespeicherten Informationen zu.

Als erstes möchten wir dazu wieder die Kundennummer benutzen. Mit Hilfe der Funktion

ergebnis=SEEKKEY(dateinummer,
Sperrparameter,
Tabellennummer,
Schlüsselwert)

können wir Jetsam veranlassen, in einer bestimmten Indextabelle nach einem Schlüsselwert zu suchen. Für den Sperrparameter muß, wie auch bei allen folgenden Suchfunktionen, der Wert 0 eingesetzt werden. Also:

ergebnis=SEEKKEY(#1,0,kindex,
MKUK$
(kundennr.))

Ist **ergebnis=0**, dann wurde die Kundennummer in der Tabelle gefunden und die aktuelle Position (s.o.) entsprechend eingestellt. Nun kann die Adresse mit **GET #1** in den Satzpuffer gelesen und über die Feldvariablen angesprochen werden. Bei **GET** ist die Angabe einer Satznummer **nicht** notwendig, da Jetsam diese, anhand der in der Indextabelle gespeicherten Information, schon kennt.

Wird von der Funktion der Wert 103 zurückgegeben, dann existiert in der Tabelle weder die gesuchte noch eine größere Kundennummer. Auch befinden sich in allen anderen Indextabellen mit höheren Tabellennummern keine Schlüsselwerte.

Der Wert 105 zeigt ebenfalls an, daß die Kundennummer in der Tabelle nicht gefunden wurde. Es können aber

1. in der Tabelle noch größere Kundennummern vorhanden sein,
oder
2. in Tabellen mit höheren Tabellennummern weitere Schlüsselwerte vorhanden sein.

Das Suchen einer bestimmten Kundenadresse nach dem Namen oder der Postleitzahl stellt sich dagegen als etwas schwieriger heraus, da in den entsprechenden Indextabellen ein Schlüsselwert mehrfach auftreten darf. Haben wir z.B. mehrere Kunden mit dem Namen **Meier**, so wird von Jetsam nach Aufruf der **SEEKKEY**-Funktion mit

ergebnis=SEEKKEY(#1,0,nindex,
"Meier")

die aktuelle Position auf den ersten **Meier** lautenden Eintrag gestellt. Handelt es sich bei dieser Kundenadresse nicht um die gesuchte, so müssen irgendwie alle anderen noch evtl. vorhandenen Meiers nach der gewünschten Adresse durchsucht werden. Man müßte nun zum nächsten Schlüssel in der Indextabelle gehen und feststellen, ob dieser den gleichen Wert, also **Meier**, hat.

Wenn ja, kann nach Laden der dazugehörigen Adresse mit **GET** und Untersuchung des Vornamens oder durch Anzeige auf dem Bildschirm entschieden werden, ob es sich um den gesuchten Kunden handelt oder nicht. Dazu bietet Jetsam die zwei Funktionen **SEEKNEXT** (suche Nachfolger) und **SEEKPREV** (suche Vorgänger) zum »Durchblättern« der Indextabellen an.

Die Syntax lautet für beide

ergebnis=SEEK????(Dateinummer,
Sperrparameter)

oder

ergebnis=SEEK????(Dateinummer,
Sperrparameter,
Tabellennummer,
Schlüsselwert,
Satznummer)

(für die Fragezeichen ist entsprechend **NEXT** oder **PREV** einzusetzen)

In der ersten Form wird von der augenblicklich aktuellen Position, die sich z.B. nach der **ADDREC-**, **ADDKEY**- oder einer Suchfunktion ergeben hat, ausgegangen. Bei der zweiten Form wird die Position, von der aus Jetsam gehen soll, durch die drei Parameter **Tabellennummer**, **Schlüsselwert** und **Satznummer** bestimmt. So können Such-

funktionen, die diese Form erlauben, immer von einem durch den Benutzer bestimmten Datum aus gestartet werden, wenn die Position durch die **FETCH????**-Funktionen abgefragt worden war.
Die in **ergebnis** zurückgegebenen Werte sollen nun anhand der **SEEKNEXT**-Funktion erläutert werden. Wird als Ergebnis der Wert 0 zurückgegeben, so hat der neue gefundene Schlüssel den gleichen Wert, wie der vorhergehende, d.h. wir haben es wieder mit einem **Meier** zu tun und müssen entsprechend handeln (s.o.)
Ist dagegen **ergebnis=101**, so hat der gefundene Schlüssel einen anderen Wert, z.B. Müller, wir haben also den gesuchten **Meier** nicht gefunden.



Wird der Wert 102 ausgegeben, so befindet sich der gefundene Schlüssel in der nächsten, mit Werten versehenen, Indextabelle, in unserem Fall handelt es sich um eine Postleitzahl in der Tabelle 2 **pindex**.
In allen drei Fällen wird die aktuelle Position auf den entsprechenden Eintrag gestellt, d.h. mit **GET** wird die zum gefundenen Schlüsselwert gehörende Adresse geladen.
Erhält man als Ergebnis den Wert 103, so war der vor dem Funktionsaufruf durch die aktuelle Position adressierte Schlüssel der letzte Eintrag der letzten benutzten Tabelle, in unserem Fall die letzte Postleitzahl in der Postleitzahlentabelle, oder anders ausgedrückt, der letzte Schlüssel der gesamten Indexdatei. Jetsam kann nun keine neue aktuelle Position bestimmen und diese ist daher undefiniert.
Der Wert 105 zeigt an, daß bei der Verwendung der zweiten Form eine falsche Position angegeben wurde. Das bedeutet, daß in der Tabelle kein Schlüsselwert mit dieser Satznummer existiert. In dem Fall wird von Jetsam die aktuelle Position auf den Schlüssel in der Indextabelle gesetzt, der unmittelbar dem gewünschten folgen würde. Existiert dieser auch nicht, d.h. die neue Position wäre hinter dem letzten Schlüssel in der Indexdatei, so ist die aktuelle Position unbestimmt.

Schließlich kann auf dem Joyce noch der Wert 115 zurückgegeben werden, der anzeigt, daß die Funktion nicht ausgeführt werden kann, da die aktuelle Position undefiniert ist und im Funktionsaufruf auch keine Positionsangabe gemacht wurde.
Die Ergebniswerte der **SEEKPREV**-Funktion entsprechen den oben aufgeführten, allerdings mit dem Unterschied, daß mit 102 der Übergang zur vorhergehenden Indextabelle und mit 103 der erste Schlüssel in der ersten Indextabelle angezeigt wird.

Wie zu sehen ist, kann mit diesen beiden Funktionen bequem in der gesamten Indexdatei, in einer Tabelle oder in einem Teilbereich einer solchen geblättert werden. Sollen z.B. alle Adressen der Kunden Namens Meier ausgedruckt werden, so braucht man nur den ersten mit **SEEKKEY** suchen, mit **GET** laden und ausdrucken. Danach fährt man mit der **SEEKNEXT**-Funktion so lange fort, bis sich als Ergebnis ein Wert größer 0 einstellt, der mit 101 einen neuen Schlüsselwert, mit 102 das Verlassen der Namen-Indextabelle oder mit 103 das Ende der Indexdatei anzeigt.

Beispiel:

```
ergebnis=SEEKKEY( #1,0,nindex,
                   "Meier")
WHILE ergebnis=0
    GET # 1
    drucken
    ergebnis=SEEKNEXT( #1,0)
WEND
```

Sollen dagegen alle Adressen von **Meier** bis **Schmidt** gedruckt werden, ist folgendes Programm notwendig:

```
ergebnis=SEEKKEY( #1,0,nindex,
                   "Meier")
WHILE ergebnis < 102 AND
FETCHKEY$( #1) < ="Schmidt"
    GET # 1
    drucken
    ergebnis=SEEKNEXT(   1,0)
WEND
```

Wollen wir uns z.B. alle benutzten Kundennummern ansehen, dann können wir die **SEEKKEY**-Funktion zum Einstellen der aktuellen Position auf den ersten Schlüsselwert der Indextabelle nicht verwenden, da wir möglicherweise die momentan kleinste Kundennummer nicht wissen. Deshalb kann dies mit der **SEEKRANK**-Funktion vorgenommen werden:

```
ergebnis=SEEKRANK(dateinummer,
                  Sperrparameter,
                  Tabellennummer)
```

Wenn **ergebnis=0** ist, konnte die aktuelle Position auf den ersten Schlüsselwert in der angegebenen Tabelle eingestellt werden.

Bei **ergebnis=102** ist in der geforderten Tabelle kein Schlüssel vorhanden und die Position wird auf den ersten Eintrag der nächsten benutzten Tabelle gesetzt. Existiert keine solche, dann wird **ergebnis=103** ausgegeben.

Das Programm dazu:

```
ergebnis=SEEKRANK( #1,0,kindex)
WHILE ergebnis < 102
    PRINT CVUK(FETCHKEY$( #1))
    ergebnis=SEEKNEXT( #1,0)
WEND
```

Interessant könnte auch eine Liste der Postleitzahlen sein, um zu sehen, in welchen Gebieten wir als Händler noch keine Kunden haben und deshalb dort die Werbetrommel ein wenig heftiger rühren müssen. Allerdings würde dann bei obigem Verfahren ein und dieselbe Postleitzahl, je nach Anzahl der Kunden in diesem Ort, mehrfach ausgegeben – wo einmal genügen würde!

Auch daran wurde bei Jetsam gedacht. Mit

```
ergebnis=SEEKSET(Dateinummer,
                 Sperrparameter)
```

kann von der aktuellen Position aus, der nächste **abweichende** Schlüsselwert gesucht werden. Ebenfalls kann, wie bei **SEEKNEXT/SEEKPREV**, die zweite Form verwendet werden. Auch die in Ergebnis zurückgegebenen Werte und ihre Bedeutung entsprechen denen der **SEEKNEXT**-Funktion, mit Ausnahme des Wertes 0, der von **SEEKSET** nicht erzeugt wird.

Das Programm zum Auflisten der Postleitzahlen sieht dann so aus:

```
ergebnis=SEEKRANK( #1,0,pindex)
WHILE ergebnis <102
    PRINT FETCHKEY$( #1)
    ergebnis=SEEKSET(# 1,0
WEND
```

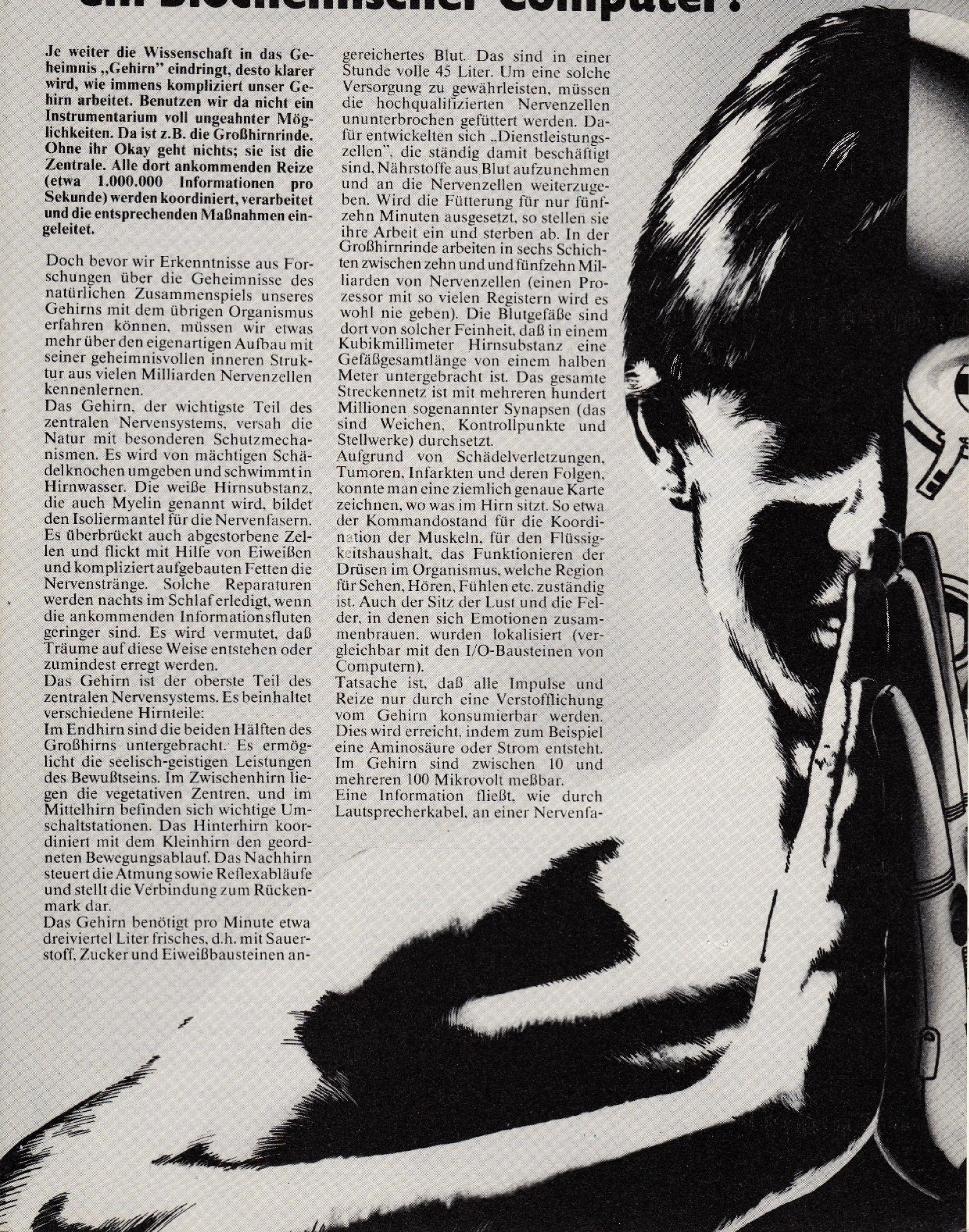
Wie zu sehen ist, bietet Jetsam mit Hilfe der **SEEK**-Funktionen, neben dem einfachen Auffinden von Daten, auch eine Vielzahl von Möglichkeiten zur sequentiellen Verarbeitung einer indizierten Datei. Sortierte Tabellen lassen sich einfach erstellen mit auf- oder absteigender Reihenfolge, Begrenzung auf Teilbereiche usw.

Der Vollständigkeit halber soll auch nicht unerwähnt bleiben, daß mit Jetsam gespeicherte Daten in einer indizierten Datei geändert oder wieder gelöscht werden können. Wer sich die Geldausgabe für ein Datenbankprogramm ersparen möchte und lieber Zeit in die Programmierung einer eigenen kleinen, auf seine Bedürfnisse zugeschnittenen verwenden möchte, hat damit ein sehr gutes Werkzeug zur Verfügung.

Mit Hilfe des Basic-Handbuches, den Anregungen aus diesem Artikel, sowie etwas Ausdauer, könnten dabei sehr interessante Programme entstehen.

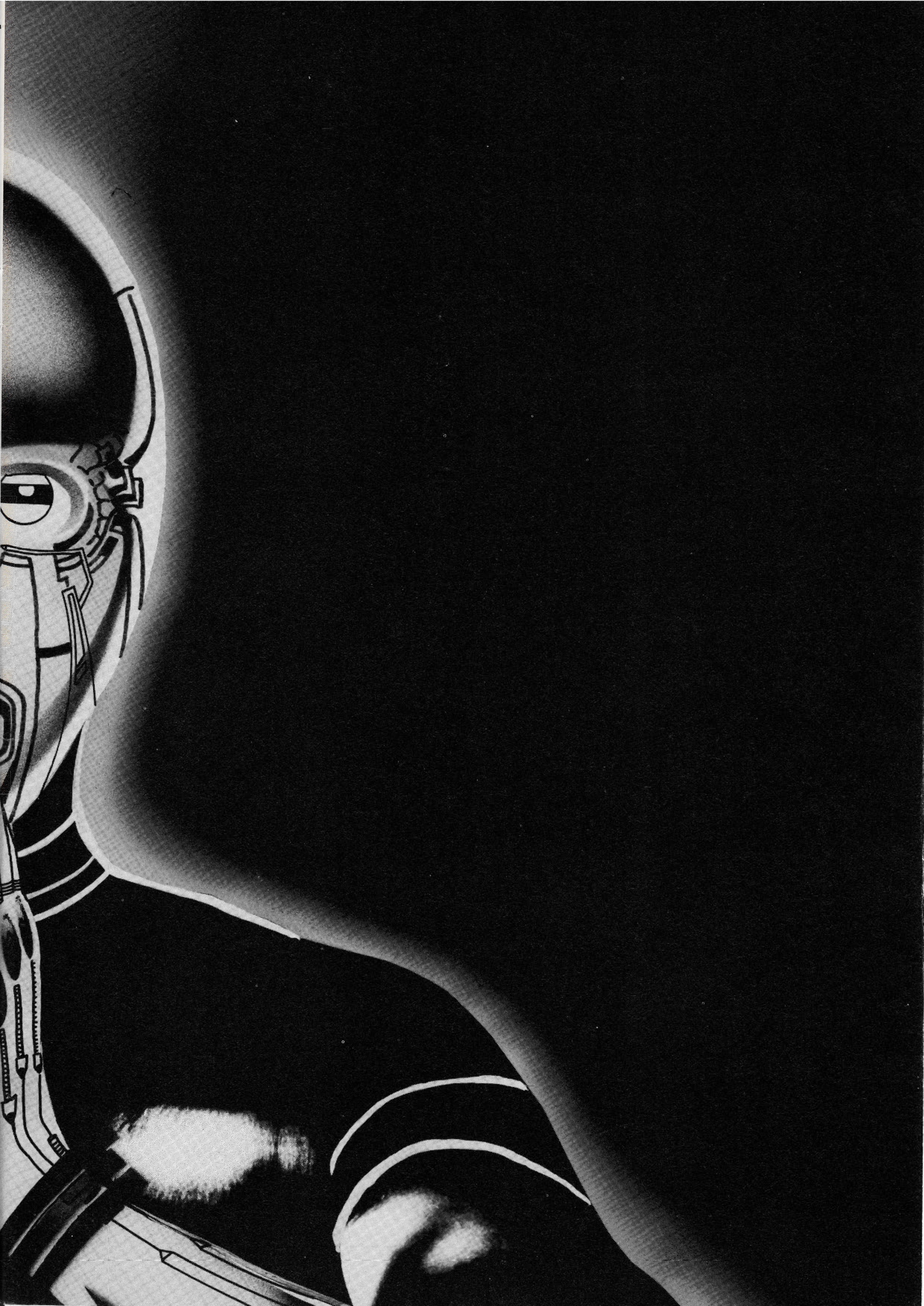
(MC)

GEHIRN
– ein biochemischer Computer?



Je weiter die Wissenschaft in das Geheimnis „Gehirn" eindringt, desto klarer wird, wie immens kompliziert unser Gehirn arbeitet. Benutzen wir da nicht ein Instrumentarium voll ungeahnter Möglichkeiten. Da ist z.B. die Großhirnrinde. Ohne ihr Okay geht nichts; sie ist die Zentrale. Alle dort ankommenden Reize (etwa 1.000.000 Informationen pro Sekunde) werden koordiniert, verarbeitet und die entsprechenden Maßnahmen eingeleitet.

Doch bevor wir Erkenntnisse aus Forschungen über die Geheimnisse des natürlichen Zusammenspiels unseres Gehirns mit dem übrigen Organismus erfahren können, müssen wir etwas mehr über den eigenartigen Aufbau mit seiner geheimnisvollen inneren Struktur aus vielen Milliarden Nervenzellen kennenlernen.
Das Gehirn, der wichtigste Teil des zentralen Nervensystems, versah die Natur mit besonderen Schutzmechanismen. Es wird von mächtigen Schädelknochen umgeben und schwimmt in Hirnwasser. Die weiße Hirnsubstanz, die auch Myelin genannt wird, bildet den Isoliermantel für die Nervenfasern. Es überbrückt auch abgestorbene Zellen und flickt mit Hilfe von Eiweißen und kompliziert aufgebauten Fetten die Nervenstränge. Solche Reparaturen werden nachts im Schlaf erledigt, wenn die ankommenden Informationsfluten geringer sind. Es wird vermutet, daß Träume auf diese Weise entstehen oder zumindest erregt werden.
Das Gehirn ist der oberste Teil des zentralen Nervensystems. Es beinhaltet verschiedene Hirnteile:
Im Endhirn sind die beiden Hälften des Großhirns untergebracht. Es ermöglicht die seelisch-geistigen Leistungen des Bewußtseins. Im Zwischenhirn liegen die vegetativen Zentren, und im Mittelhirn befinden sich wichtige Umschaltstationen. Das Hinterhirn koordiniert mit dem Kleinhirn den geordneten Bewegungsablauf. Das Nachhirn steuert die Atmung sowie Reflexabläufe und stellt die Verbindung zum Rückenmark dar.
Das Gehirn benötigt pro Minute etwa dreiviertel Liter frisches, d.h. mit Sauerstoff, Zucker und Eiweißbausteinen angereichertes Blut. Das sind in einer Stunde volle 45 Liter. Um eine solche Versorgung zu gewährleisten, müssen die hochqualifizierten Nervenzellen ununterbrochen gefüttert werden. Dafür entwickelten sich „Dienstleistungszellen", die ständig damit beschäftigt sind, Nährstoffe aus Blut aufzunehmen und an die Nervenzellen weiterzugeben. Wird die Fütterung für nur fünfzehn Minuten ausgesetzt, so stellen sie ihre Arbeit ein und sterben ab. In der Großhirnrinde arbeiten in sechs Schichten zwischen zehn und und fünfzehn Milliarden von Nervenzellen (einen Prozessor mit so vielen Registern wird es wohl nie geben). Die Blutgefäße sind dort von solcher Feinheit, daß in einem Kubikmillimeter Hirnsubstanz eine Gefäßgesamtlänge von einem halben Meter untergebracht ist. Das gesamte Streckennetz ist mit mehreren hundert Millionen sogenannter Synapsen (das sind Weichen, Kontrollpunkte und Stellwerke) durchsetzt.
Aufgrund von Schädelverletzungen, Tumoren, Infarkten und deren Folgen, konnte man eine ziemlich genaue Karte zeichnen, wo was im Hirn sitzt. So etwa der Kommandostand für die Koordination der Muskeln, für den Flüssigkeitshaushalt, das Funktionieren der Drüsen im Organismus, welche Region für Sehen, Hören, Fühlen etc. zuständig ist. Auch der Sitz der Lust und die Felder, in denen sich Emotionen zusammenbrauen, wurden lokalisiert (vergleichbar mit den I/O-Bausteinen von Computern).
Tatsache ist, daß alle Impulse und Reize nur durch eine Verstofflichung vom Gehirn konsumierbar werden. Dies wird erreicht, indem zum Beispiel eine Aminosäure oder Strom entsteht. Im Gehirn sind zwischen 10 und mehreren 100 Mikrovolt meßbar.
Eine Information fließt, wie durch Lautsprecherkabel, an einer Nervenfa-



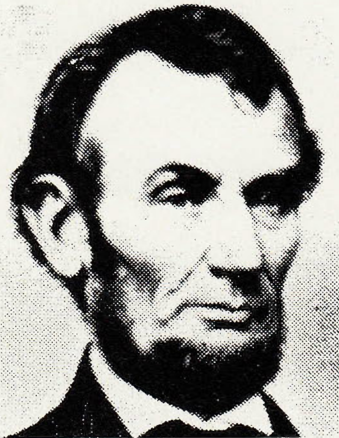
ser entlang. Als Transportbehälter dienen dabei Eiweißkörperchen. Bei der ungeheuren Vielfalt von Verknotungen der Fasern (Neuriten), stößt dieses Päckchen nach Millisekunden an Kreuzungen. Dort werden mittels chemischer Überträgersubstanzen, den Transmittern, die anliegenden Synapsen abgefragt, ob Interesse an der Sendung besteht. Nur derjenige, der informationshungrig ist, gibt ein entsprechendes Zeichen, so daß der Informationsfluß weiterlaufen kann (bei Computern wird dies Handshake genannt). Billionen von Synapsen spielen so mit Milliarden von Nervenzellen 'Frage und Antwort' - ununterbrochen; in kaum faßbaren Kombinationen.

Das Gedächtnis

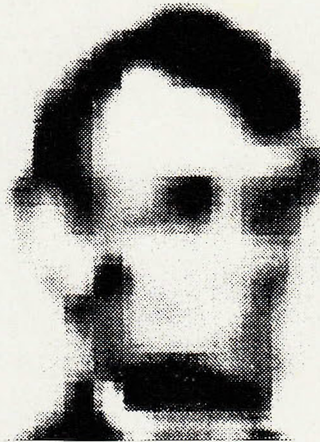
Um sich etwas für längere Zeit merken zu können, muß das Gehirn bzw. die Großhirnrinde auf chemischem Weg mit Hilfe von Eiweißbausteinen eine Matrix herstellen. Dies dauert etwa zwanzig Minuten. Um bei Ausfall eines Gehirnteils nicht die gesamte Erinnerung zu verlieren, wird diese „Kopie" einer Erinnerung zerstückelt und auf verschiedene Datenbanken verteilt. Zur Verdeutlichung:
Stellen Sie sich vor, Sie speichern ein Bild ab. Dann beinhaltet jedes Teilstück das gesamte Bild; allerdings mit verringerter Schärfe.

BEISPIEL: Bilderinnerung

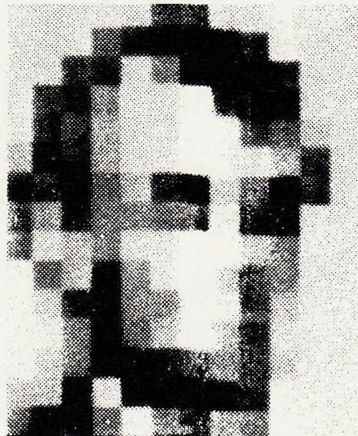
vollständige Erinnerung



ein Teil verloren



zwei Teile verloren



Doch wird nicht alles abgespeichert, denn eine solche große Kapazität dürfte selbst für ein Gehirn zu groß sein. Hier bedient sich das Gehirn eines Tricks: Elemente des Gedächtnisses.

Eine ankommende Information wird nicht einfach entweder behalten oder vergessen, sondern sie verweilt in drei unterschiedlich langen Speicherstufen.

Das Ultrakurzzeitgedächtnis

Alle durch die Sinnesorgane ankommenden Impulse kreisen zunächst einmal in Form elektrischer Ströme und Schwingungen in unserem Gehirn, wo sie nach zehn bis zwanzig Sekunden wieder abklingen. Wenn keine Aufmerksamkeit vorhanden ist oder wenn sich diese Information nicht an den bereits bekannten Gedankenverbindungen aufhängen läßt, werden diese Informationen wieder vergessen.

Das Kurzzeitgedächtnis

So kurz die Verweilzeit in unserem Ultrakurzzeitgedächtnis auch sein mag, sobald wir die Information innerhalb dieser paar Sekunden abrufen oder wenn sie irgendwelchen bereits gespeicherten Gedächtnisinhalten zugeordnet werden können und eine Resonanz mit schon vorhandener Erinnerung erzeugt wird, können auch die neuen Eindrücke vor dem Verlöschen gerettet werden. Hier werden diese Informationen dann für etwa zwanzig Minuten gespeichert.

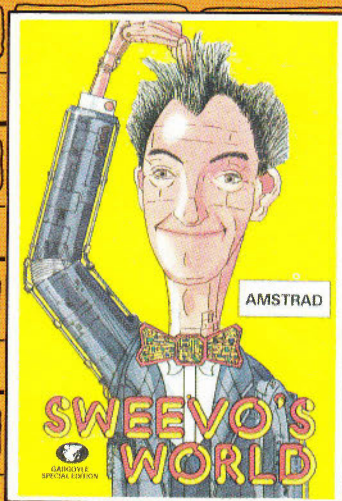
Das Langzeitgedächtnis

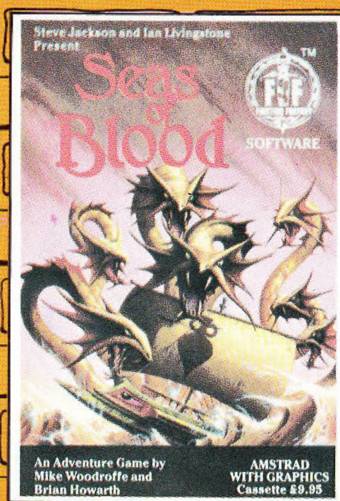
Wird auf eine Information während der Speicherung im Kurzzeitgedächtnis zugegriffen, so wird, wie oben schon erwähnt, eine Matrix hergestellt. Was so verarbeitet ist, bleibt bei uns als sogenanntes Langzeitgedächtnis unter günstigen Umständen ein ganzes Leben lang gespeichert.

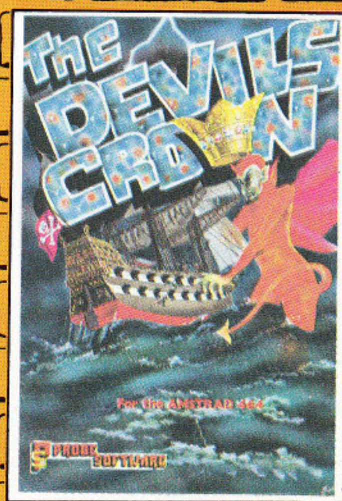
Doch fast alles bis jetzt beschriebene könnte auch ein Computer erledigen. Aber was unterscheidet ein Gehirn von einem Computer? Es ist die schöpferische Tätigkeit, das kreative Element unseres vielfältigen Gehirns - und damit genau die Tätigkeit, zu der ein Computer (noch) nicht fähig ist. Und doch entstehen schöpferisches Denken, Einfälle und Intuitionen aus stofflichen, also materiell gespeicherten Informationen. Diese Code-Moleküle sind kreuz und quer über der ganzen Großhirnrinde verteilt. Und nur aus diesem Grund ist es möglich zu assoziieren und zu kreieren - d.h., aus vielen Informationen völlig neue Sinnesinhalte und damit eine schöpferische Idee zu erzeugen. (HF)

Inserentenverzeichnis









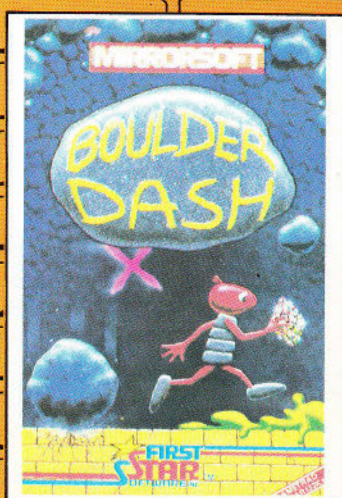


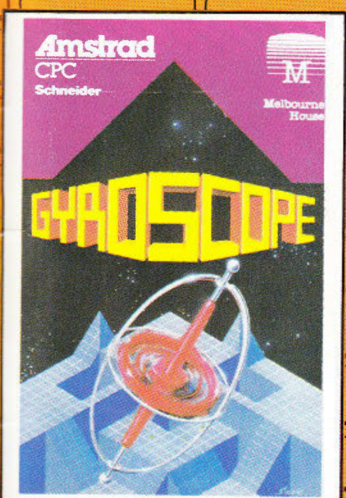
und hier weitere Spitzen-Spiele:

Art.-Nr.	Titel	Verk. Preis
136 C	Tank Busters	29,— DM
137 C	Dark Star	29,— DM
538 C	Spitfire 40	39,— DM
539 C	Skyfox	35,— DM
540 C	Cylu	14,90 DM
541 C	Theatre Europe	39,— DM
542 C	Super Sleuth	35,— DM
218 C	Elite (deutsch)	59,— DM
543 D	Dun Darach	55,— DM
544 D	Marsport	55,— DM
545 D	Yie ar Kung Fu	55,— DM
546 C	Desert Rats	39,— DM
547 C	Thing	35,— DM
549 D	Code Name Mat 2	59,— DM
550 D	Working backwards	59,— DM
551 D	Hypersports	49,— DM
552 D	The Amstr. Music System	79,— DM
553 D	Mini Office 2 (deutsch)	79,— DM
554 D	Macrocosmica	55,— DM

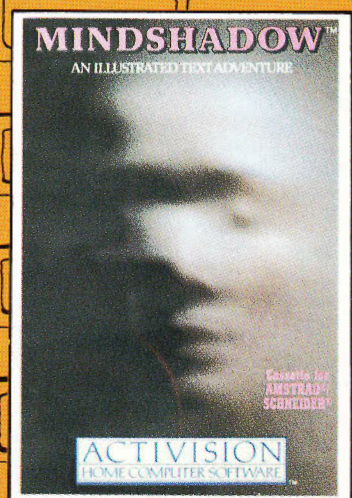
Zu den Abbildungen:

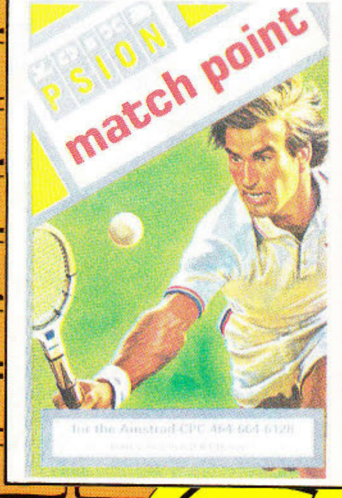
Art.Nr.	Titel	Verk. Preis
514 C	Technician Ted	29,90 DM
503 C	Sweevo's World	29,90 DM
504 C	Seas of Blood	45,— DM
521 C	The Devils Crown	45,— DM
515 C	Boulder Dash	45,— DM
– C	Match Point	39,— DM
522 C	Mindshadow	45,— DM
524 C	Starstrike	29,90 DM
517 C	Gyroscope	39,— DM
501 C	Trange Loop	39,— DM

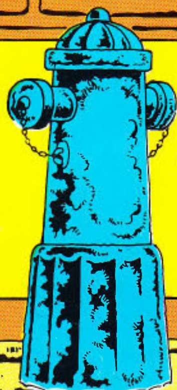














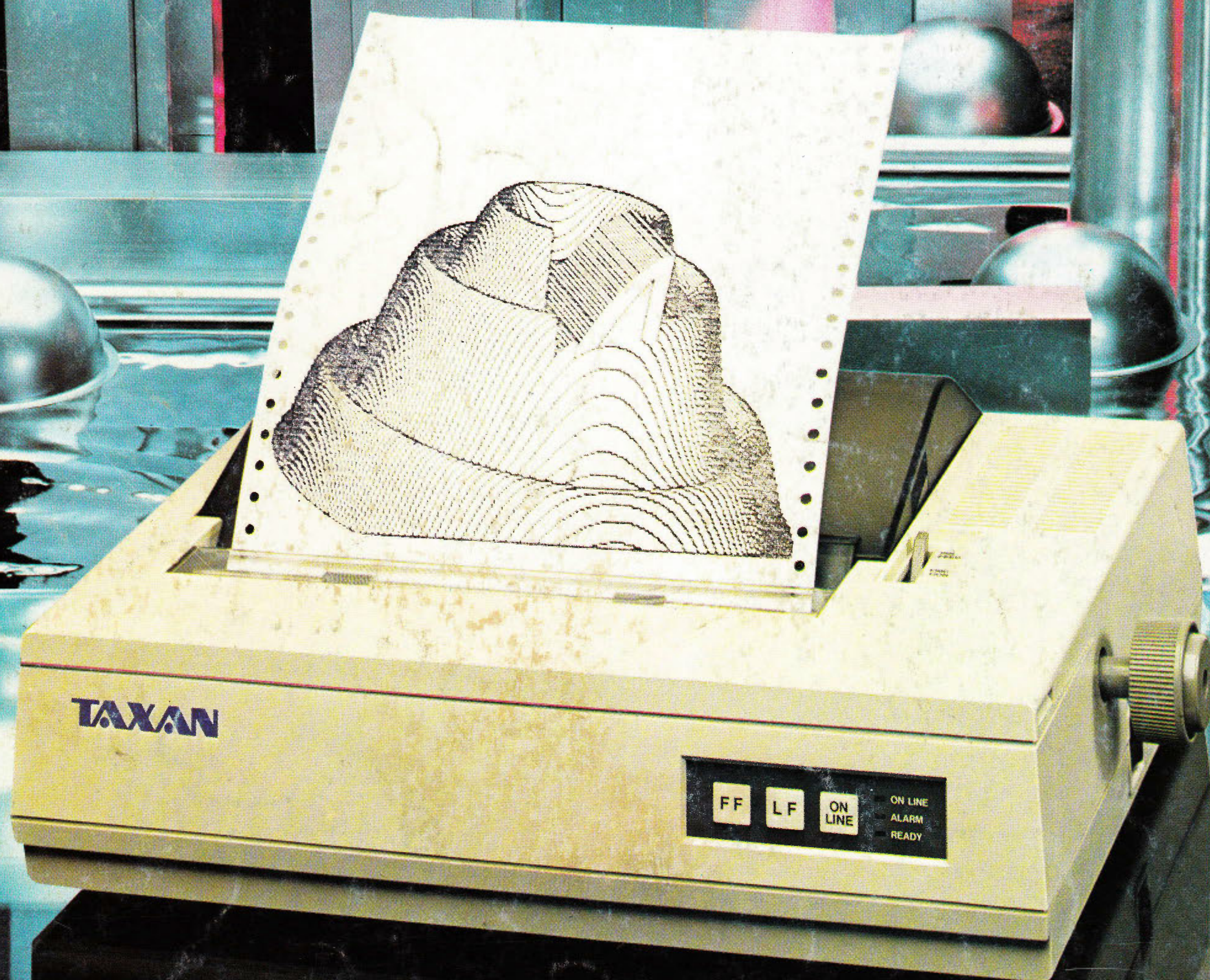
Data Media MAILORDER

Bestellen Sie noch heute, Lieferung erfolgt umgehend! Zahlung per Vorkasse oder Nachnahme zuzügl. Porto- bzw. Nabhnahmegebühr (Nachnahme ins Ausland ist nicht möglich).

Weitere Artikel in unserem Gesamtkatalog. Bitte anfordern (2,– DM für Rückporto beilegen).
Data Media GmbH -Mailorder- Ruhrallee 55, 4600 Dortmund, Tel.: (02 31) 12 50 71-3

TAXAN – für die Welt von morgen.

Hannover Messe '86
CeBIT —— 12.–19. 3. 1986
Halle 17, Stand 315/414



TAXAN KP-810/910: professioneller Matrixdrucker. Schnell: 140 Zeichen/Sek. · leise: < 60 dB · Schönschrift: Near Letter Quality (NLQ) · variabel: frei programmierbarer Zeichensatz · ausbaufähig: Zusatzkarten mit verschiedenen Schriftarten/IBM PC-Zeichensatz. KP 810 – 80 Zeichen pro Zeile. KP 910 – 156 Zeichen pro Zeile. TAXAN – denn die Zukunft hat bereits begonnen.

Ausführliche Information erhalten Sie beim Fachhändel.